BASIC para la escuela

nivel 1



Decoroso Crespo López
Luis Manuel Valdez González

BASIC PARA LA ESCUELA
(Nivel I)

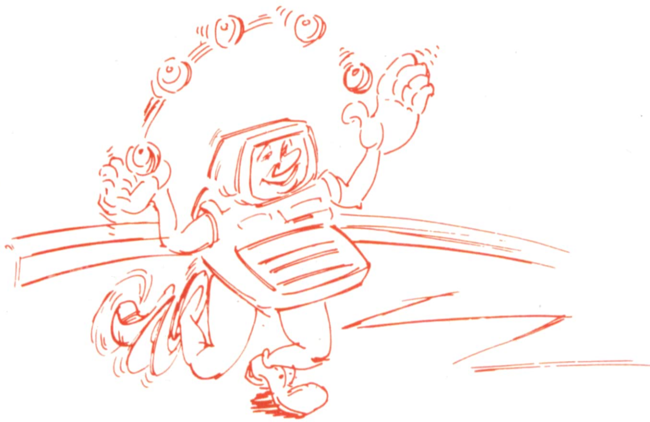
Centro de Orientación de Sociología y Psicología Aplicadas
Madrid-1984

LUIS MANUEL VALDEZ GONZALEZ
Licenciado en Informática, Profesor de Formación Profesional.

DECOROSO CRESPO LOPEZ
Adjunto de Informática Teórica de la
Facultad de Informática de Madrid.

EDITA: **COSPA** - Juan Antonio Martín
2.ª Edición 1985
Depósito Legal: GU-530-1984
ISBN: M-398-2178-6
Edición Reservada de Cospa
C/. Bravo Murillo, 377, 6º (Plaza de Castilla) - Teléf. 733 20 89
28020 MADRID (España)
Imprime: P y Punto - Tel. 656 05 86
San Fernando de Henares (Madrid)

PROLOGO



COSPA ha iniciado, en cuanto Empresa de Servicios Mecanizados para la Enseñanza, una nueva actividad con la que pretende que los conocimientos básicos de la informática sean impartidos en los Centros de Educación de EGB, BUP, COU y FP.

A este nuevo Servicio le denominamos **AULA DE INFORMATICA BASICA,** y constará de tres niveles:

Nivel 1: "Introducción a los ordenadores y Programación en Lenguaje BASIC".

Nivel 2: que constará de las tres siguientes alternativas opcionales:

2.1.— Perfeccionamiento de la programación en Lenguaje BASIC (Segundo curso).

2.2.— Iniciación a la programación en el Lenguaje Educacional LOGO (Primer curso).

2.3.— Iniciación a la programación en el Lenguaje PASCAL (Primer curso).

Nivel 3: igualmente constará de las tres alternativas opcionales siguientes:

3.1.— Programación en Lenguaje BASIC (Tercer curso).

3.2.— Perfeccionamiento de la programación en Lenguaje LOGO (Segundo curso).

3.3.— Perfeccionamiento de la programación en Lenguaje PASCAL (Segundo curso).

Pues bien, **BASIC PARA LA ESCUELA** (nivel 1) responde a los objetivos a conseguir por el ***AULA DE INFORMATICA BASICA*** *en su primer nivel. Es decir, por un lado asimilar los conocimientos básicos del ordenador y su uso, tanto de la máquina (Hardware) en sí, como de los procedimientos de programación (Software); por otro, asimilar los conocimientos elementales de la programación en Lenguaje BASIC.*

Los alumnos, aprendiendo la lógica del funcionamiento del computador y la lógica de la programación misma, conseguirán simultáneamente desarrollar su inteligencia adquiriendo el hábito de pensar estructuralmente. Ello le servirá sin duda, para perfeccionar su modo de estudiar las demás áreas de aprendizaje.

Por otro lado, y como objetivo no menos importante que los anteriores, el ***AULA DE INFORMATICA BASICA*** *pretende que, con el dominio del lenguaje de Programación BASIC (así como los demás lenguajes programados), que se inicia con este nivel elemental de* ***BASIC PARA LA ESCUELA,*** *los alumnos estén capacitados para la utilización de* ***LA***

ENSEÑANZA ASISTIDA POR ORDENADOR *con suficiente autonomía y mejor aprovechamiento.*

La metodología utilizada en la elaboración de este Manual Elemental ***BASIC PARA LA ESCUELA*** *es la que hemos considerado mejor para conseguir una didáctica adaptada y de fácil asimilación para los alumnos a los que va dirigido. La complejidad va creciendo progresivamente conforme se avanza en el estudio, pero permitiendo realizar prácticas desde el primer día.*

Todos los capítulos del Manual guardan una misma estructura interna. En primer lugar se explica el concepto con sencillez y claridad. Se completa la explicación mediante ejemplos adecuados a cada punto. Al final de cada capítulo se repasan los conceptos tratados en el mismo mediante preguntas y/o ejercicios que el alumno debe saber responder sin necesidad de recurrir a la explicación de texto. El alumno podrá contrastar sus respuestas con las soluciones correctas que el Manual le ofrece.

Finalmente, se han elaborado una serie de ejercicios sobre cuestiones de Matemáticas y de Física que deberán resolverse mediante la elaboración del correspondiente programa en Lenguaje BASIC. De esta se conseguirá afianzar los conocimientos adquiridos sobre el lenguaje BASIC aplicándoles en la resolución de unos problemas que han estudiado en otras áreas del aprendizaje. Evidentemente, los programas a realizar para resolver dichas cuestiones sólo necesitan de los conceptos elementales del BASIC estudiado en este Manual.

Estos ejercicios se presentan en el Apéndice, en el que la primera Parte consta de problemas de Matemáticas y en la segunda Parte se presentan las cuestiones de Física.

Para cada uno de los ejercicios se presenta una de las soluciones correctas posibles. Es decir, un programa en lenguaje BASIC que resuelve de forma correcta cada una de las cuestiones planteadas, de forma que el alumno pueda contrastar su solución con la propuesta y revise los errores cometidos analizando sus causas.

Por tanto, querido lector, creemos haber puesto en sus manos un Manual elemental de BASIC que está elaborado pensando en aquellas personas que carecen de conocimientos previos en este lenguaje, de forma que le sirva de una ayuda metodológicamente valiosa y didáctica para adentrarse en el conocimiento de la programación mediante el lenguaje BASIC.

L. Ortega M.

Director del Departamento de Educación e Investigación de COSPA

Madrid, septiembre de 1984

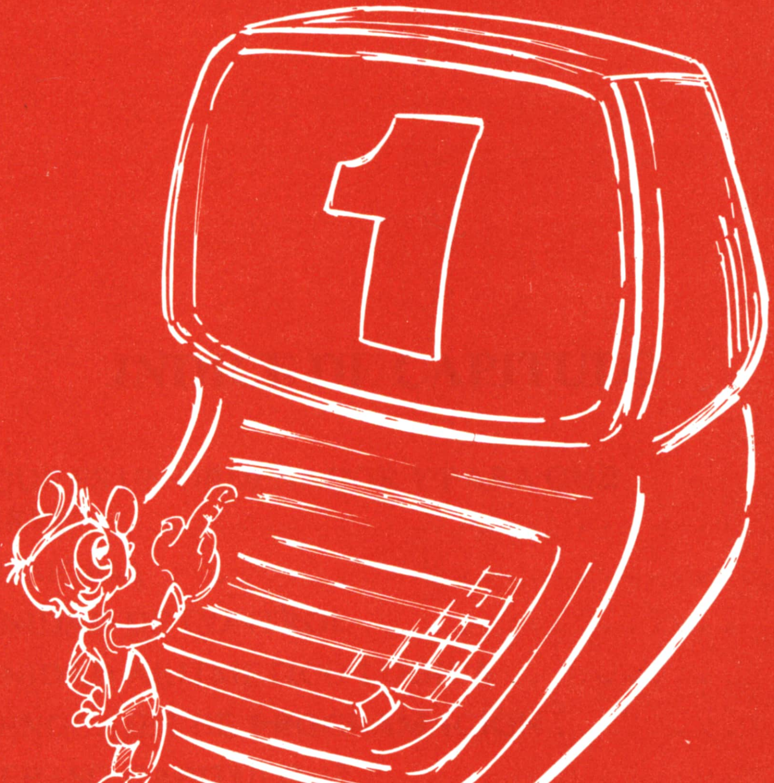
INDICE GENERAL

CAPITULO 4º. INSTRUCCIONES BASIC.

CAPITULO 5º. BUCLES Y TRANSFERENCIAS DE CONTROL

CAPITULO 6º. LISTAS Y TABLAS.

APENDICE


1

INDICE DE CAPITULO

CAPITULO 1º

EL ORDENADOR



1. ¿QUE ES UN ORDENADOR Y PARA QUE SIRVE?

Un ordenador es una máquina que ejecuta automáticamente una serie de operaciones (instrucciones) que se han dado previamente.

Posiblemente, esta definición no te diga gran cosa, por eso, para entenderlo mejor, lo veremos con un ejemplo:

imaginaros una empresa que tiene que pagar a sus empleados, necesita calcular el dinero que tiene que entregar a cada uno de ellos. Para esto, tiene que utilizar datos como sueldo base, antigüedad, hijos, bonificaciones, retenciones, Seguridad Social, etc. (datos de entrada).

A todos estos datos hay que aplicarles una serie de operaciones y en un determinado orden (programa), para que nos calcule el dinero que hay que darle a cada empleado (resultado).

Evidentemente, si en esta empresa hubiese 3 empleados, lo podríamos hacer a mano o con la ayuda de una calculadora, pero si la empresa tuviese 1.800 empleados la cosa se complicaría, pues tardaríamos mucho o necesitaríamos muchas personas para hacer estos cálculos. Sin embargo, para el ordenador resultaría más o menos sencillo y sobre todo, mucho más rápido, ya que la velocidad a la que el ordenador ejecuta los cálculos es infinitamente más rápida que la del hombre.

Para que un trabajo de este tipo se pueda realizar en un ordenador, necesitamos darle las siguientes informaciones:

DATOS DE ENTRADA, serán aquellos a partir de los cuales vamos a realizar los cálculos (sueldo base, antigüedad...).

PROGRAMA O DATOS OPERACIONALES, tenemos que decirle al ordenador las operaciones que tiene que hacer con los datos de entrada y en qué orden se deben hacer. Además, se le debe decir en un lenguaje que entienda el ordena-

dor (lenguajes de programación como son por ejemplo el BASIC, FORTRAN, COBOL, etc.).

Y el ordenador nos dará:

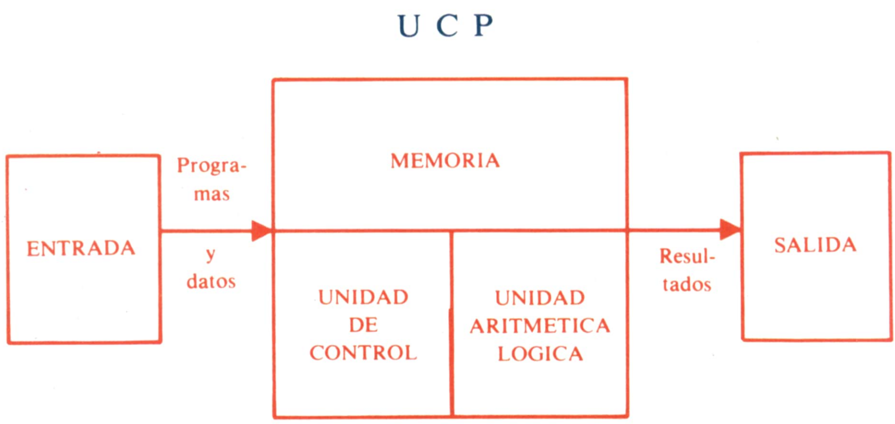
RESULTADOS O DATOS ELABORADOS, es decir, el resultado de haber aplicado las operaciones a los datos de entrada.

Una vez visto lo que hace un ordenador, vamos a ver cómo lo hace. Para ello vamos a descomponerlo en las partes de que consta.

Todo ordenador consta de dos partes principales:

1. LA UNIDAD CENTRAL DE PROCESO (UCP), en la que se acumulan los datos y programas y se realiza el tratamiento de la información en general.

2. LAS UNIDADES PERIFERICAS, que nos permiten la comunicación con el ordenador, es decir, introducir datos u obtener resultados del ordenador.



LA UNIDAD CENTRAL DE PROCESO se compone de:

— Memoria, donde se guarda la información que usa el ordenador.

— Unidad Aritmético Lógica, que se encarga de realizar las operaciones necesarias con la información.

— Unidad de Control, que se encarga de dirigir a las demás.

2. MEMORIA

En la memoria del ordenador sólo se pueden guardar dos tipos de información básica, el 0 y el 1, es decir, podemos suponer la memoria como una serie de puntos que pueden estar encendidos, 1; o apagados, 0.

La unidad de información es el "bit", y sólo tiene dos valores 0 y 1.

Así pues, la información que se guarda en la memoria solo puede estar compuesta por unos y ceros.

Podemos imaginarnos la memoria como una libreta de páginas numeradas, en cada página hay una serie de cuadrículas y en cada cuadrícula podemos guardar un cero o un uno.

En cada página podemos escribir una palabra, y el tamaño de la palabra depende del número de cuadrículas (bits) que tiene cada página, esto a su vez depende del tipo de ordenador, hay ordenadores que tienen 8, 16, 32 bits (cuadrículas) por palabra (página).

En un ordenador determinado, todas las páginas tienen el mismo número de cuadrículas; o lo que es lo mismo, una palabra siempre tiene el mismo numero de bits.

Hemos visto que en la memoria del ordenador se puede guardar información, pero ¿qué información se puede guardar?. Bien, pues en la memoria se guardan **Instrucciones** y **Datos**. Instrucciones que indican al ordenador qué operaciones debe realizar y datos con los cuales debe realizar estas operaciones.

El ordenador necesita saber qué Instrucción debe realizar y con qué datos debe de hacerlo, para eso necesita saber dónde están y poder obtenerlos. Por eso decíamos que las páginas de la memoria están numeradas, para que el ordenador pueda acceder a esa información (Instrucción o Dato) sabiendo solamente el número de página donde se encuentra.

Es decir, el ordenador se puede encontrar instrucciones del tipo:

Pon en la página 10 el valor 100
Ejecuta la instrucción de la página 32

Los datos pueden ser valores numéricos o bien palabras que estarán guardados en memoria en forma de ceros y unos, según unos códigos preestablecidos.

La capacidad de una memoria es el número de páginas que tiene y se suele expresar en K.

Si a un grupo de 8 bits lo llamamos byte u octeto, una K son 1.024 bytes, es decir, 1.024 grupos de 8 bits.

Luego una memoria de 64 kbytes quiere decir que tiene 64 × 1.024 bytes, o bien, 64 × 1.024 × 8 bits.

3. UNIDAD DE CONTROL

Es la parte del ordenador encargada de controlar la realización de las operaciones, y el orden de las mismas. Así como de dar paso a otras partes del ordenador como la UAL.

¿Cómo lo hace?

a) En el ordenador hay un apuntador que en todo momento contiene la posición o número de página de la instrucción que se debe de ejecutar, este apuntador se conoce con el nombre de Contador de posiciones.

b) Entonces, la Unidad de Control va a la posición o página que le indica el contador de posiciones y obtiene la instrucción.

c) Lee la instrucción y la interpreta o decodifica, es decir, decide qué es lo que indica esa instrucción.

d) Si la operación es una operación aritmética o lógica, le dice a la Unidad Aritmético Lógica qué tiene que hacer y con qué datos.

e) Si la operación es una operación de entrada/salida, el control se lo comunica a la parte del ordenador que se

dedica a realizar estas operaciones (unidad de E/S, canal) para que se ejecute.

4. UNIDAD ARITMETICO LOGICA

Los ordenadores disponen de una parte que específicamente se dedica a realizar operaciones aritméticas, (sumas, restas, multiplicaciones, etc...) y lógicas, (comparaciones...).

Para esto, la Unidad Aritmético Lógica dispone de unos circuitos, sumadores, comparadores, etc., que toman como entrada los datos que le da la unidad de Control; y mediante los circuitos lógicos seleccionados, de acuerdo con la operación que se va a realizar, devuelve el resultado de la operación que indicaba la instrucción que previamente había interpretado la U.C.

5. LENGUAJES DE PROGRAMACION

Para que el ordenador haga las operaciones necesarias con los datos de entrada y nos dé los resultados que nosotros deseamos, debemos decírselo de alguna forma que el ordenador entienda; es decir, en un lenguaje que sea comprensible para el ordenador.

Por ejemplo, si quisiéramos sumar dos números podríamos decirle:

Entrar número A
Entrar número B
Sumar A y B y ponerlo en C
Imprimir C
Fin

Este lenguaje, que para nosotros es comprensible, debemos hacerlo comprensible para el ordenador, así pues:

```
10 INPUT "Introduce el primer Numero:  ";A :Entrar numero A
20 INPUT "Introduce el segundo Numero: ";B :Entrar numero B
30 LET C=A+B                              :Sumar A y B y ponerlo en C
40 PRINT C                                :Imprimir C
50 END                                    :Fin
```

Este es un lenguaje que comprende el ordenador pero, al igual que ocurre con los idiomas, hay varios lenguajes comprensibles a los ordenadores, siendo los más conocidos BASIC, FORTRAN, COBOL, PASCAL, RPG...

Resumiendo, un programa es una serie de instrucciones escritas en un lenguaje comprensible para el ordenador.



REPASO CAPITULO 1º

P1) El computador esencialmente se compone de dos partes que son:

a) ....

b) ....

P2) La Unidad Central de Proceso se compone de 3 partes:

1) Una parte llamada .... donde se guarda la información que usa el computador.

2) Una parte llamada .... que hace las operaciones necesarias con la información.

3) Una parte llamada .... que dirige a las demás.

P3) ¿Cómo se llama la unidad de información que sólo puede contener como valor un cero o un uno?

....

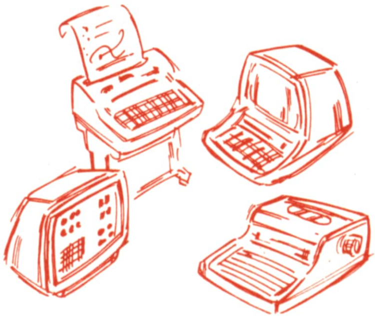
P4) ¿Cómo se llaman las unidades mediante las cuales nosotros podemos darle la información al computador y mediante las cuales el computador nos da los resultados?

....

STOP

No pasar la página hasta haber contestado las cuatro preguntas planteadas.

SOLUCIONES



R1) a) La U.C.P. (Unidad Central de Proceso)
b) Las Unidades Periféricas o unidades de entrada/salida

R2) 1) Memoria
2) Unidad Aritmético-Lógica
3) Unidad de Control

R3) BIT

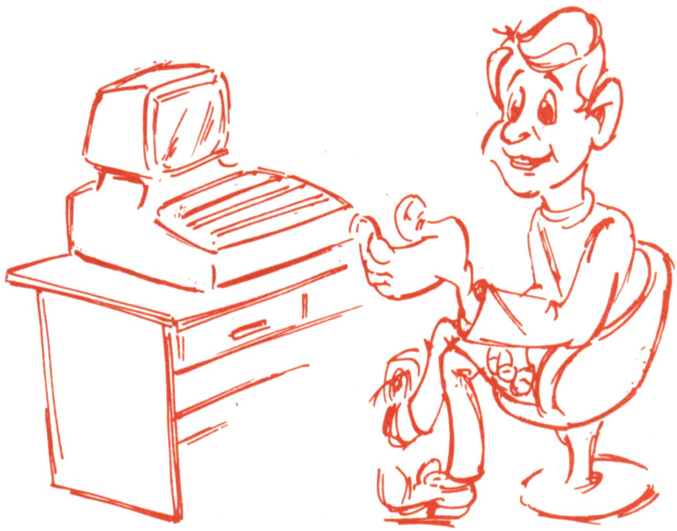
R4) Unidades Periféricas o unidades de entrada/salida


2

INDICE DE CAPITULO

CAPITULO 2º

ORDENADORES PERSONALES

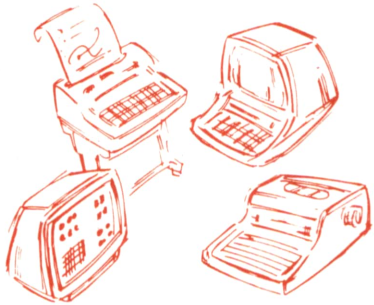


1. INTRODUCCION

Los ordenadores personales son de pequeña dimensión y precio relativamente bajo, con unas prestaciones limitadas, pero más amplias que los ordenadores medios de hace algunos años.

Los periféricos más normales que tienen estos ordenadores son:

TECLADO (entrada)
PANTALLA (salida)
IMPRESORA (salida)
CASSETTE (entrada/salida)



2. TECLADO

Suele ser el método más corriente para introducir datos al ordenador. Son teclados muy similares a los de una máquina de escribir, con alguna tecla más para funciones especiales o de control.

3. PANTALLA

Como periférico de salida es muy versátil, permite la salida de información en un aparato de televisión o un Monitor, pueden ser en blanco y negro o color. En ellos se pueden escribir todos los caracteres y símbolos especiales y, en algunos casos, gráficos de media y alta resolución.

4. IMPRESORA

Los ordenadores pueden tener como periférico de salida una pequeña impresora, de papel normal o térmico, que nos

servirá para imprimir programas o resultados de los cuales necesitamos tener constancia escrita.

5. CASSETTE

Es una unidad de almacenamiento externo que nos sirve para poder guardar programas de una forma prolongada y poder utilizarlos cuantas veces sean necesarias sin necesidad de volver a introducirlos desde el teclado.

6. HISTORIA DEL ORDENADOR

Desde el ábaco hasta nuestros días, la informática ha evolucionado de forma sorprendente. Como primer antecedente de calculadora digital se puede citar la máquina de Pascal en 1642, que sumaba y restaba números enteros por medio de combinaciones de ruedas dentadas. Esta máquina fue perfeccionada por Leibnitz en 1671, incluyendo la posibilidad de multiplicar y dividir.

No obstante, como precursores de los ordenadores modernos podemos citar a:

• CHARLES BABBAGE, profesor de Matemáticas de la Universidad de Cambridge, que desarrolló una máquina para resolver ecuaciones de 2.° grado, aprovechando la idea de utilización de fichas perforadas para la entrada de datos en su máquina.

• H. AIKEN, que de 1937 a 1944 desarrolló el llamado Mark-I, calculador automático esencialmente de tipo mecánico.

• J.P. EAKERT y J.W. MAUCHLY, desarrollaron hacia 1945, en la Universidad de Pensilvania, el ENIAC, utilizando componentes y circuitos electrónicos y unas 18.000 válvulas electrónicas.

El primer ordenador comercial fue el UNIVAC I, que apareció en el mercado en 1951. Fue el primero en utilizar técnicas binarias. También por esta época, IBM lanzó sus modelos 600 y 700. Los ordenadores de esta época son los llamados de la primera generación y todos utilizaban válvulas de vacío, ocupaban un gran espacio y producían mucho calor.

Con el desarrollo del transistor como sustitutivo de las válvulas electrónicas y su aplicación a los ordenadores, nace la segunda generación. Estos ordenadores ocupan mucho menos volumen y gastan menos energía, con lo que disminuye el calor producido, su fiabilidad es mayor y el mantenimiento más sencillo.

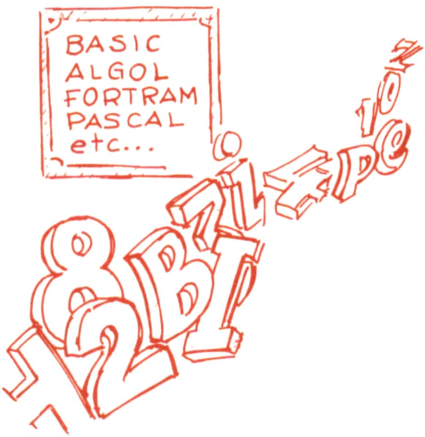
Esta generación se puede considerar iniciada entre 1955 y 1960.

El desarrollo y perfeccionamiento del transistor, unido a las técnicas de miniaturización de los circuitos electrónicos y de sus componentes, (tales como diodos, resistencias, condensadores) dan paso a la tercera generación con los circuitos inte-

grados monolíticos de pequeñísimo volumen, gran fiabilidad y enorme flexibilidad de diseño y funcionalidad.

Los ordenadores actuales se construyen a base de circuitos integrados con una gran densidad de componentes. Esto hace que se hable ya de la cuarta generación.

7. GLOSARIO



ALFANUMERICO

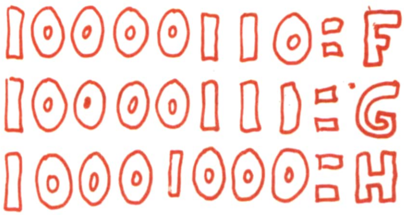
Información que contiene caracteres **alfabéticos** (letras); **números** o/y **caracteres especiales** (—,; + /).

BIT

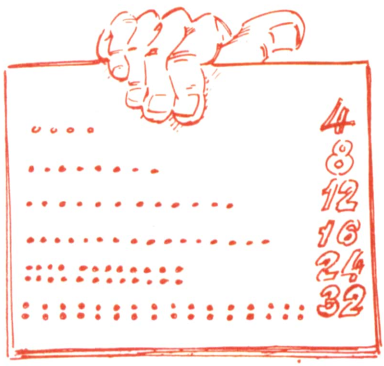
(Binary-unit), unidad de información binaria, que **sólo** puede contener o cero o uno.



BYTE U OCTETO



Grupo de 8 bits. Se puede referir también al espacio necesario en la memoria del ordenador para guardar un carácter.



PALABRA

Número de bits mínimo que el ordenador maneja a la vez.

Los valores más normales son: 4, 8, 12, 16, 24, 32.

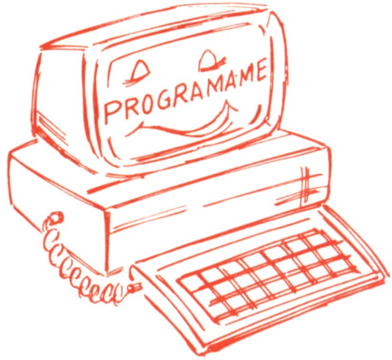
INSTRUCCION

Componente de un programa que le indica al ordenador **la operación a realizar y los datos** con qué realizarla, en un lenguaje comprensible al ordenador.



PROGRAMA

Lista de instrucciones que le dicen al ordenador los cálculos a realizar y el orden en que debe hacerlo.





DATOS

Información necesaria sobre la cual el ordenador debe realizar las operaciones. Pueden ser valores numéricos, alfanuméricos...

INPUT, ENTRADA

Operación de **entrada** de datos a la memoria del ordenador desde un periférico de entrada, teclado...



OUTPUT, SALIDA

Operación consistente en la **extracción** de resultados del ordenador en un periférico de salida, pantalla, impresora...



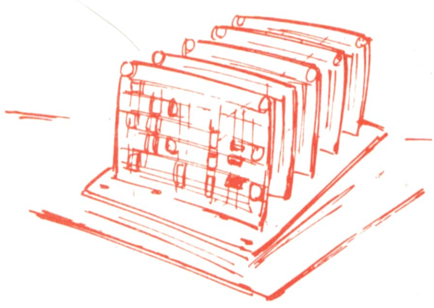


SOFTWARE

Conjunto de programas formado por nuestros programas de aplicación y por el núcleo de programas que hacen funcionar al ordenador (Sistema Operativo).

HARDWARE

Equipo físico que compone el ordenador, es todo aquello del ordenador que podemos tocar.

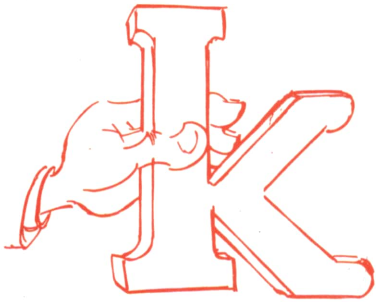


CHIP



Combinación de **muchos componentes electrónicos** en una placa cuadrada de silicio de unos 4 mm. aproximadamente.

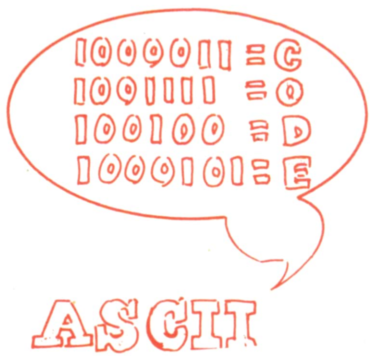
K



Unidad de medida de capacidad de memoria. Una K corresponde a 1.024 bytes. Una memoria de 2 K's será de 2.048 bytes.

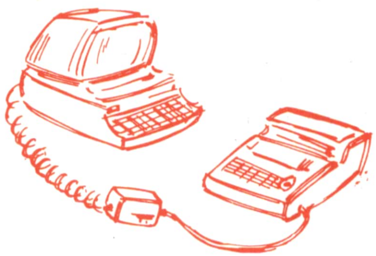
ASCII (American Standard Code for Information Interchange).

Código Standar Americano para el intercambio de información. Es un código de 7 bits y resulta ser el más utilizado en ordenadores personales. A cada letra o carácter le corresponde una combinación diferente de 7, 0'S y 1'S.



INTERFACE

Digamos que es el **enchufe físico para adaptar dos sistemas**; por ejemplo, para conectar una impresora al ordenador necesitamos un interface, que físicamente será el cable que une a los dos.





MICROPROCESADORES

Es lo mismo que la CPU, pero adopta este nombre debido al reducido tamaño que tienen estos procesadores (CPU).

MICRO-ORDENADOR

Un ordenador completo que consta de:

- — un microprocesador
- — memoria
- — unidades de E/S, (periféricos).



RAM (Random Acces Memory)

Memoria de acceso aleatorio, es la memoria del ordenador en la que puedo **leer y grabar datos** y programas.



EPROM (Erasable Programable Read Only Memory)

Cumple la misma función que un Rom, pero con algo más; puedes **"borrar"** la información que habías grabado previamente, usando luz ultravioleta, y **grabarla de nuevo** con otra información.



ROM (Read Only Memory)

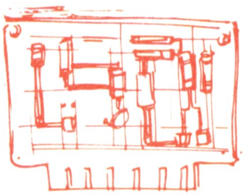
Es un tipo de memoria del ordenador que está grabada con **información fija** y en la que sólo se pueden hacer operaciones de lectura. **"Nunca"** se olvida de su información y **no puede cambiarse.**



PROM (Programable Read Only Memory)

Es una memoria ROM, con la diferencia de que se fabrican **sin grabar nada,** y quien va a usar esta memoria escribirá lo que quiera en ella utilizando equipos especiales. Una vez escritas **no se puede cambiar su contenido**.



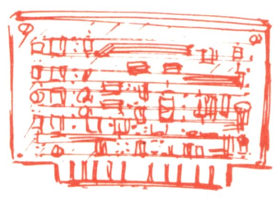


TARJETA

Es el **soporte de los circuitos impresos** y componentes. Suele ser una placa de material aislante.

BUS

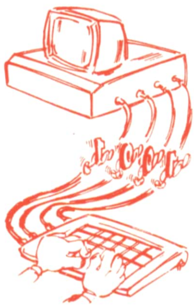
Es **la conexión** a través de la cual se mueven los datos e instrucciones en el ordenador.



E/S SERIE

La transmisión en serie consiste en **mandar la información bit a bit**.

Se suele hacer cuando las distancias son largas.



BUFFER

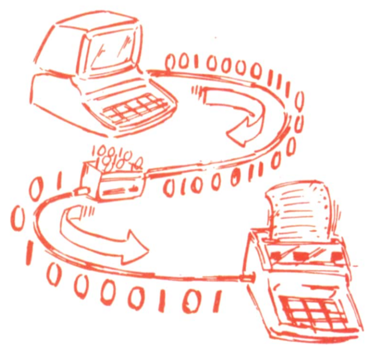
(O Memoria Intermedia) es una **zona de memoria** que se utiliza **para adecuar operaciones que tienen distinta velocidad**. Por ejemplo, si estuviésemos sacando información a la impresora, es más rápido el proceso que la velocidad de impresión, entonces el buffer retiene y dosifica esta información para que no se pierda.



E/S PARALELO

Se **transmiten a la vez tantos bits como tenga la palabra del ordenador**.

Esta modalidad es más rápida, pero también más cara y se utiliza en distancias cortas.



MEMORIA EXTERNA

Son dispositivos, normalmente magnéticos, donde podemos **guardar grandes lotes de información** para su posterior utilización.





CASSETTE

En un ordenador personal podemos guardar información en un cassette. Este es uno de los dispositivos magnéticos antes mencionados.

FLOPPY DISK

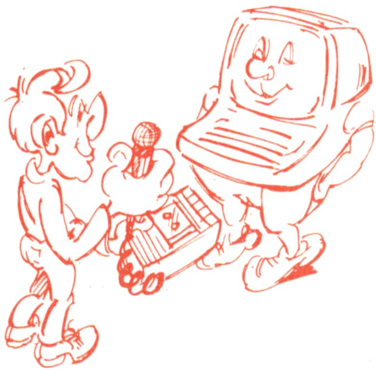
Este es otro de los dispositivos magnéticos utilizados como memoria externa en los ordenadores personales. Es un disco magnético flexible que puede almacenar mucha información.



DISCO DURO



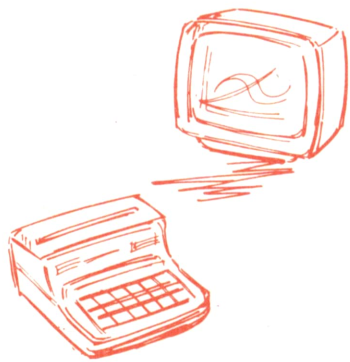
Es otro medio de almacenamiento externo, aunque no es tan usual en ordenadores personales. Puede almacenar una gran cantidad de información. Su precio es más caro que los anteriores.



TERMINAL

Es un puesto de trabajo remoto, que se comunica con el ordenador y puede introducir y obtener datos y programas del ordenador al que está conectado.

CRT (Cathode Ray Tube)



Pantalla que reproduce la imagen de la información que tiene el ordenador.

Es un periférico de salida de información.

RESUMEN

REPASO CAPITULO 2º

P1) ¿Cuáles son los periféricos más normales en un ordenador personal?

....

P2) ¿Cuál fue el primer ordenador comercial y en qué año apareció en el mercado?

....

P3) ¿Cómo se llama un grupo de 8 bits?

....

P4) Una lista de instrucciones que le dice al ordenador lo que tiene que hacer y en que orden, es un ....

P5) El equipo físico que compone el ordenador, es decir, todo aquello que se puede tocar, es el ....

P6) ¿Cuál es la medida de capacidad correspondiente a 1.024 bytes?

....

P7) Decir dos soportes magnéticos de almacenamiento.

....

STOP

No pasar la página hasta haber contestado las preguntas planteadas.

SOLUCIONES



R1) a) Teclado
b) Pantalla
c) Impresora
d) Cassette

R2) UNIVAC I
Año 1951

R3) Byte u Octeto

R4) Programa

R5) Hardware

R6) K

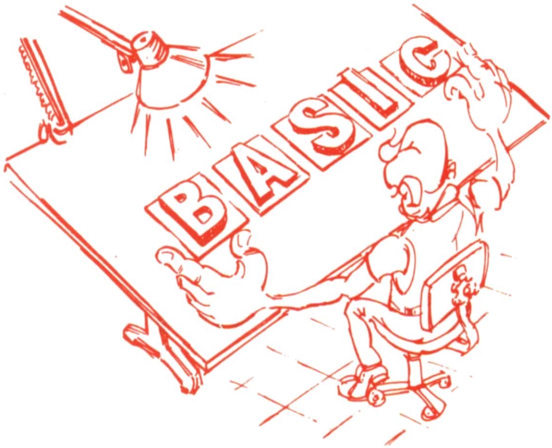
R7) Cassette
Floppy Disk


3

INDICE DE CAPITULO

CAPITULO 3º

BASIC



1. INTRODUCCION AL BASIC

BASIC son las iniciales de Beginners All-purpose Symbolic Instruction Code. Código de instrucción simbólico de uso múltiple para principiantes.

El Basic es un lenguaje de programación que se caracteriza por su sencillez y versatilidad. Es muy ágil y no responde a ninguna estructura preestablecida.

Maneja un alfabeto compuesto de letras, números y caracteres especiales.

- letras (A — Z) Mayúsculas y Minúsculas
- números (0 — 9)
- caracteres especiales (+ , —, *, /, ;, '', '', ...)

Veámos un programa escrito en Basic.

```
10 REM CALCULO DEL MAYOR DE DOS NUMEROS
20 INPUT "NUMERO A ";A
30 INPUT "NUMERO B ";B
40 IF A>B THEN PRINT "EL MAYOR ES ";A:GOTO 60
50 PRINT "EL MAYOR ES ";B
60 END
```

Aquí podemos ver que cada línea tiene un número de orden que se llamará n.º de línea; por ejemplo en la línea 50,

```
50 PRINT "EL MAYOR ES ";B
```

- Tenemos un n.º de línea: **50**
- Una instrucción: **PRINT**
- Unas expresiones: **''EL MAYOR ES ='' ; B**
- Y al conjunto de PRINT ''EL MAYOR ES ='' ; lo llamaremos sentencia.

Como podemos ver en la línea 40, dos o más sentencias pueden ir en la misma línea separadas por **dos puntos** (:).

2. CONSTANTES

Una constante puede ser un número o un conjunto de caracteres que no variarán a lo largo de todo el programa.

Hay dos tipos de constantes:

a) **Numéricas:**

El Basic admite todo tipo de números enteros o decimales, por ejemplo

124.52

0.072

37

pueden ser positivos o negativos

—378.32

—0.0032

tiene otro formato para números demasiado grandes o demasiado pequeños, que se denomina formato Exponencial o tipo E; por ejemplo:

$7.420 = 7.42 \times 10^3$

y el Basic nos lo mostrará con el siguiente formato

X.XXXX E n

por ejemplo:

74270000 el Basic lo presenta como 7.427 E 7

es decir,

$7.427\ E\ 7 = 7.427 \times 10^7 = 74270000$

b) **Alfanuméricas:**

Una constante alfanumérica es un conjunto de caracteres (números, letras, caracteres especiales) cualquiera, puesto entre comillas.

Ejemplo:

```
"JAVIER SERRA"
"1425 PESETAS"
"45 A\OS"
```

3. VARIABLES

El concepto de variable es uno de los más importantes en programación. ¿Qué es una variable?; una variable es un espacio de memoria que el ordenador reserva para guardar un valor (numérico o alfanumérico) y ese espacio tiene un nombre (nombre de variable) por el cual el ordenador lo reconoce.

Hay que diferenciar entre nombre de variable y contenido de la misma.

Por ejemplo:

```
30 LET A=25+7
```

El ordenador reserva un espacio que, a partir de esta instrucción, lo va a conocer por el nombre de **A**; y en ese espacio va a introducir el valor **32**.

Así pues, Nombre de Variable = A

contenido de la Variable = 32

Siempre que hagamos referencia a A, nos estamos refiriendo a un contenido.

```
40 LET C=A+1
50 PRINT C
```

El ordenador nos imprimirá **33,** pues al decirle que nos sume a A la constante 1, hará

```
C=32+1
```

y al decirle que nos imprima C, nos imprime su contenido 32 + 1 = 33

El nombre de una variable ha de comenzar siempre por una letra, y puede ir seguida de más letras o números.

Por ejemplo: A, A1, AA, PTS, NOM1, N1D... son nombres válidos de variables.

Hay ordenadores que sólo distinguen los 2 primeros caracteres como nombre de variable, así:

RA y RADIO, los consideraría como la misma variable.

Cuando le damos valor a una variable con un signo =, debemos de tener en cuenta, que el funcionamiento del signo no es el mismo que en Matemáticas.

Para asignarle un valor a una variable en algunos ordenadores, es necesario utilizar la instrucción LET.

Por ejemplo:

```
10 LET C=20+3
20 LET D=C+1
30 PRINT C,D
```

El ordenador lo que hace es calcular la parte derecha del igual (=), y ese valor lo asigna a la variable que está a la parte izquierda.

En el ejemplo anterior, primero calcula la parte derecha del igual de la instrucción 10, 20 + 3 = 23 y ese valor se lo asigna a la variable de nombre C.

En la instrucción 20, calcula la expresión a la derecha del igual, 23 + 1 = 24 y se lo asigna a la variable de nombre D.

Luego este programa imprimirá

```
23 24
```

Hay dos tipos de variables en relación a su contenido:

a) **Numéricas:** que sólo pueden contener valores numéricos.

b) **Alfanuméricas:** en las que se pueden guardar cualquier conjunto de caracteres (letras, números, caracteres especiales).

Para darles nombre sirven las mismas normas, pero deben llevar al final el símbolo $.

```
10 LET A$="LUIS"
20 LET C$=" 35+7"
30 PRINT A$,C$
```

Este programa imprimirá:

```
LUIS 35+7
```

4. OPERACIONES Y FORMULAS

En el lenguaje Basic, hay tres tipos de operadores:

a) **Aritméticos:** que indican las operaciones aritméticas ele-

mentales; suma (+), resta (—), multiplicación (*), división (/) y potenciación ^.

b) **De comparación:** que indican la comparación a realizar entre constantes, variables, expresiones; igual (=); menor que (<); menor o igual que (<=); mayor que (>); mayor o igual que (>=); desigual (<>).

c) **Lógicos:** que se utilizan para unir dos o más condiciones, los más importantes son:

— AND — la condición final es cierta si lo son las condiciones elementales.

— OR — la condición final es cierta si al menos es cierta una de las elementales.

— NOT — la condición general es cierta si es falsa la condición elemental, y viceversa.

Ejemplos:

A = 3
B = 7
A < 4 AND B > 5 → cierto
A = 3 AND B < 5 → falso
A > 0 OR B = 1 → cierto
A < 0 OR B > 8 → falso
NOT A = 3 → falso
NOT B < 5 → cierto

Jerarquía de los operadores aritméticos

El Basic tiene una jerarquía preestablecida según la cual evalúa las expresiones numéricas.

- **Primero se ejecutan las exponenciales ^**
- **Después las multiplicaciones y divisiones *, /**
- **Y por último las sumas y restas +, —**

Las operaciones del mismo nivel se ejecutan siempre de izquierda a derecha; y por supuesto, en la ejecución se respeta siempre la jerarquía establecida por los paréntesis.

Vamos a ver cómo se escribirían las fórmulas matemáticas en lenguaje Basic para que sean entendidas por el ordenador.

Una fórmula es una expresión formada por constantes y variables relacionadas por los operadores aritméticos. Estas fórmulas se computan siempre con los valores actuales de las variables; por lo tanto, una fórmula representa siempre un número concreto.

FORMULA NOTACION MATEMATICA	FORMULA EN BASIC
$\frac{A}{b + c}$	a/(b+c)
$a^{(b+c)}$	a ^ (b+c)
$\frac{a}{b(c+d)} + e$	a/(b*(c+d))+e
$a + \frac{b}{c+d}$	a+b/(c+d)
$(a+b)c+d^{(x+y)}$	(a+b)*c+d ^ (x+y)

FORMULA NOTACION MATEMATICA	FORMULA EN BASIC
$\sqrt{(a+b)c}$	((a+b)*c) ^ (1/2)
$a^3 + b^2$	a ^ 3 + b ^ 2



REPASO CAPITULO 3º

P1) Las instrucciones siguientes son erróneas, ¿por qué?

```
10 LET 27=B
20 LET 14*2+3=C
             10
30 LET C=-----
            3+2
```

P2) Sabiendo que las operaciones aritméticas se efectúan en un determinado orden, ¿cómo se efectuarán las siguientes operaciones dentro del ordenador?

- 3 + 4 + 15 / 3
- 6 / 3 / 2
- (4 + 5) / 2 + 18
- 16 / 2 ^ 2
- (25 + 5) * 3 / (2 + 3)

P3) Transcribir al lenguaje basic las siguientes fórmulas:

- $2a^2 + 5\,b^3 + ab$
- $\sqrt[3]{a + b}$

SIGUE

- $\frac{a\,(a + 1)}{b} + \frac{b\,(b + 1)}{a}$
- $\frac{(3a^2 + b)^3}{7 + 6c}$
- $(a + b)^2 + \frac{3\,(x + y)}{2a}$

STOP

No pasar la página hasta haber contestado las preguntas planteadas.

SOLUCIONES



R1)

```
10 LET 27=B
20 LET 14*2+3=C
```

En estos casos el error está en que las asignaciones en Basic se hacen tomando el valor de la derecha del igual y asignándoselo a la variable que está a la izquierda.

```
         10
30 LET C=-----
         3+2
```

En este caso, la expresión $\frac{10}{3 + 2}$ no está correctamente escrita en Basic, ya que sería:

```
30 LET C=10/(3+2)
```

R2)

- 3 + 4 + 15 / 3
- 3 + 4 + 5

  7 + 5

  12

- **6 / 3 / 2**

  2 / 2

  1

SIGUE

- **(4 + 5)** / 2 + 18
  9 / 2 + 18
  4,5 + 18
  22,5

- 16 / **2 ^ 2**
  16 / 4
  4

- **(25 + 5)** * 3 / (2 + 3)
  30 * 3 / **(2 + 3)**
  30 * 3 / 5
  90 / 5
  18

R3)

- $2a^2 + 5b^3 + ab$
  En Basic: 2 * a ^ 2 + 5 * b ^ 3 + a * b
- $\sqrt[3]{a + b}$
  En Basic: (a + b) ^ (1/3)
- $\dfrac{a\,(a + 1)}{b} + \dfrac{b\,(b + 1)}{a}$

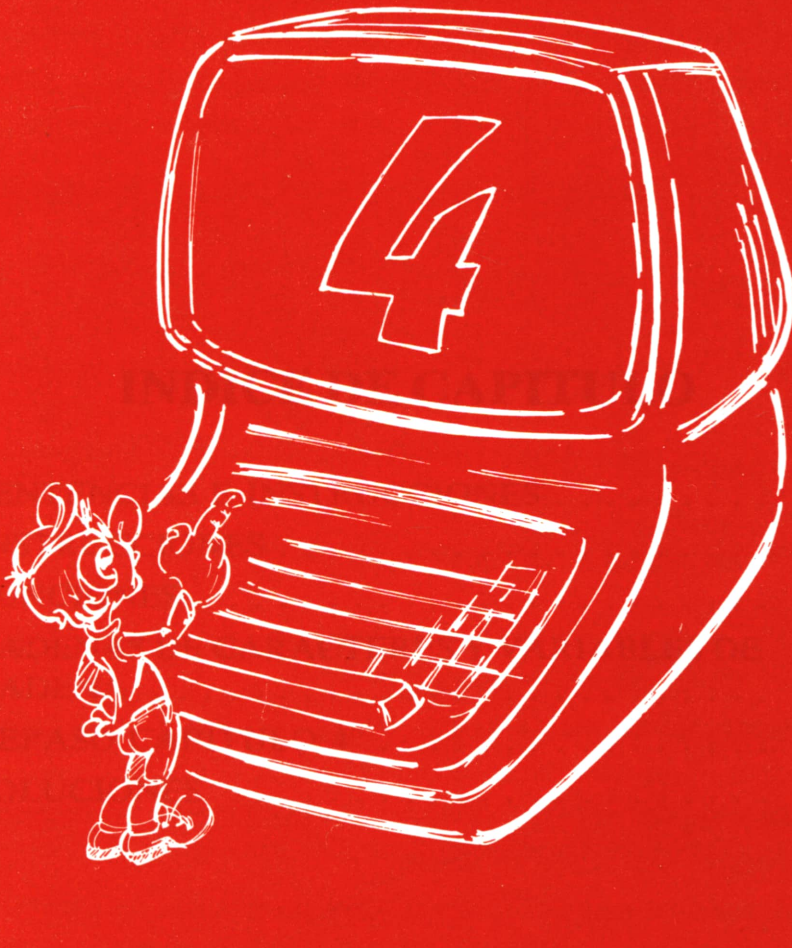
En Basic: a * (a + 1) / b + b * (b + 1) / a

- $\dfrac{(3a^2 + b)^3}{7 + 6c}$

En Basic: (3 * a ^ 2 + b) ^ 3 / (7 + 6 * c)

- $(a + b)^2 + \dfrac{3\,(x + y)}{2a}$

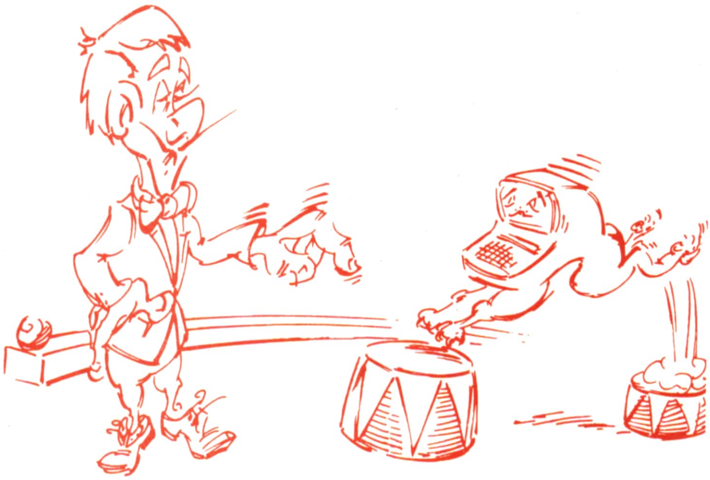
En Basic: (a + b) ^ 2 + 3 * (x + y) / (2 * a)


4

INDICE DE CAPITULO

CAPITULO 4º

INSTRUCCIONES BASIC



1. SENTENCIAS E INSTRUCCIONES

Antes de ver algunas instrucciones del Basic, conviene recordar los siguientes conceptos:

— **Instrucción:** es la parte de la sentencia que indica la operación a realizar, PRINT, LET, INPUT...

— **Sentencia:** es el conjunto de instrucción y operandos, PRINT A, B; GO TO 60; ...

— **Línea:** es una componente del programa, se compone de un número de línea y una o varias sentencias separadas por dos puntos(:).

Veámos para qué se utiliza el número que se pone al principio de cada línea de programa.

Este número le indica al ordenador el orden normal en que deben ejecutarse las instrucciones. Además, se puede apreciar en ejemplos anteriores que las líneas suelen ir numeradas de 10 en 10 y, aunque no es obligatorio, es la numeración recomendada y más usual.

Veámos qué utilidad tiene mediante un ejemplo.

```
10 REM IMPRESION DE DOMICILIO
20 PRINT "CALLE              POBLACION        TELEFONO"
30 PRINT "Lagasca, 23-2   Madrid          (91)4413729"
```

Este programa nos producirá una salida así:

```
CALLE       POBLACION       TELEFONO
LAGASCA     MADRID          (91)4413729
```

Si quisiéramos que la cabecera nos saliese subrayada, podríamos introducir la siguiente instrucción:

```
25 PRINT "________________________"
```

con lo cual, el programa nos dará el siguiente resultado:

```
CALLE       POBLACION       TELEFONO
________________________________________
LAGASCA     MADRID          (91)4413729
```

Es decir, podemos intercalar instrucciones en un programa introduciéndolas con el número de línea correspondiente al lugar que queremos que ocupe.

2. INSTRUCCIONES

REM: COMENTARIOS

Es bastante frecuente la necesidad de manejar programas elaborados por otras personas, o elaborados por uno mismo, pero bastante tiempo atrás.

Si no se han puesto líneas de comentarios a lo largo de todo el programa nos será muy difícil hacer un seguimiento del programa a través de un listado, por eso es recomendable el uso de la instrucción REM.

n.º de línea REM (comentario)

La instrucción REM no se ejecuta, es decir, el Basic la ignora, la única finalidad que tiene es la documentación de listados.

Ejemplo:

```
10 REM PROGRAMA SUMA DOS NUMEROS
20 REM AUTOR:Javier Serra
30 REM FECHA: 14-Agosto-1984
40 INPUT A
50 INPUT B
```

SIGUE

```
60 LET C=A+B : REM SUMA DE A y B
70 PRINT "La suma de A y B es = ";C
```

En algunos ordenadores la instrucción REM se puede sustituir por:

— un signo de admiración (!)

— un apóstrofe o comilla simple (')

END : FIN

Está será la última sentencia del programa que se ejecutará y hace que el ordenador de por finalizado el programa, quedando dispuesto para realizar otro trabajo cualquiera.

En algunos ordenadores no es obligatoria su utilización pero nosotros la utilizaremos y la pondremos al final de nuestros programas.

n.° de línea END

```
10 REM PROGRAMA DEL AREA DEL CUADRADO
20 INPUT "Entrar el lado= ";L
30 LET A=L*L
40 PRINT "El area del cuadrado es = ";A
50 END
```

LET : ASIGNACION

Esta instrucción se utiliza para asignarle un valor a una variable:

Tiene el siguiente formato:

n.º línea LET Variable = expresión

El ordenador, cuando se encuentra una instrucción LET, calcula el valor de la expresión y se lo asigna a la variable.

Por ejemplo:

```
100 LET A=25*10/2
```

Una vez ejecutada esta sentencia, A tendrá el valor de 125. En algunos ordenadores, esta instrucción no es necesaria.

Ejemplos:

```
10 REM PROGRAMA EJEMPLO UNO
20 LET A$="Buenos dias "
30 LET B$="JUAN: "
40 PRINT B$;A$
50 END

JUAN: Buenos dias
```

```
10 REM PROGRAMA EJEMPLO DOS
20 REM POTENCIAS DE DOS
30 LET A=2*2
40 LET B=A*2
50 LET C=B*2
```

SIGUE

```
60 PRINT A" ";B" ";C
70 END

4 8 16
```

INPUT : ENTRADA DE DATOS POR TECLADO

Formato:

n.º línea INPUT "MENSAJE"; Variable

Esta instrucción al ser ejecutada presenta en pantalla el "mensaje" y solicita al operador que le introduzca por medio del teclado un valor numérico o alfanumérico que le será asignado a la variable.

Dentro de una sentencia INPUT el mensaje es opcional. Es decir, puede ponerse o no, es más, hay ordenadores que no permiten el mensaje dentro del INPUT, esto se podría subsanar utilizando una sentencia PRINT, por ejemplo:

```
PRINT "MENSAJE";
INPUT Variable.
```

esto sería similar al formato anterior, en el caso de que la instrucción INPUT no admitiese un mensaje.

También admite más de una variable en la misma instrucción, separadas por comas, como consecuencia; a la hora de introducir los valores debemos hacerlo en el mismo orden y cantidad que las variables.

Ejemplos:

```
10 REM PROGRAMA EJEMPLO INPUT
20 INPUT "Entrar nombre: ";NO$
30 INPUT "Entrar edad:    ";E
40 INPUT "Entrar Calle y Poblacion: ";C$,P$
50 PRINT NO$" ";E;" a|os"
60 PRINT C$
70 PRINT P$
80 END
```

La ejecución de este programa producirá:

LUIS 29 AÑOS
CRUZ
GERONA

ENTRAR NOMBRE? LUIS [RETURN]
ENTRAR EDAD? 29 [RETURN]
ENTRAR CALLE Y POBLACION? CRUZ, GERONA
[RETURN]

El operador o usuario del ordenador tecleará los datos que están a la derecha del ?, si en algún caso al teclear el dato nos equivocamos de tipo (es decir, que el dato no corresponde con el tipo de variable) nos dará un mensaje de error ''REDO FROM START'' y volverá a pedirnos el dato.

Por ejemplo, si en lugar de darle 29 en la edad le diésemos **veintinueve**, nos daría dicho error, pues en una variable numérica no podemos introducir valores **alfanuméricos.**

Si este ordenador no permitiése la utilización de mensaje en el INPUT, podríamos sustituir las instrucciones como sigue:

```
20 PRINT "Entrar nombre: ";
25 INPUT NO$
```

PRINT : PRESENTACION DE RESULTADOS

Esta instrucción se utiliza para imprimir textos, variables o expresiones.

Normalmente la instrucción PRINT provoca la salida de datos en la pantalla del ordenador; la instrucción para la salida de datos por impresora (si disponemos de ella) es LPRINT, las normas son comunes para las dos.

Su formato es:

(n.º de línea) PRINT (expresiones y/o variables y/o textos)

Podemos imprimir:

- Variables

```
100 PRINT A$, B, C$
```

En cuyo caso saldrán en pantalla los contenidos de las variables.

- Constantes y expresiones

```
100 PRINT "LUIS", 25, 32 + 4
```

En cuyo caso el resultado será la impresión del valor constante o el resultado de la expresión:

LUIS 25 36

Las constantes alfanuméricas deben ir entre comillas ('').

• Una línea en blanco

100 PRINT

Esta instrucción imprime una línea en blanco; es decir, no imprime nada pero provoca un espacio entre líneas.

Cuando se escriben varios términos dentro de una sentencia PRINT, estos términos deben estar separados por coma(,) punto y coma(;) o TAB.

— **Coma:** cuando en una sentencia PRINT los términos están separados por comas, el ordenador divide la línea de impresión en unas zonas de igual longitud (normalmente 4) e imprime cada término en una de las zonas.

100 PRINT "LUIS", "JAVIER", "JOSE"

LUIS	**JAVIER**	**JOSE**
zona 1	**zona 2**	**zona 3**

— **Punto y coma:** si los términos van separados por punto y coma (;), el ordenador imprime todos, uno a continuación de otro, **sin dejar espacios en blanco**. Si uno de los términos fuese un número positivo, aparecerá un espacio en blanco antes del número que corresponde al signo. Si una sentencia PRINT acaba con un (;) la siguiente impresión se hará en la misma línea a continuación de la última impresión.

— **TAB:** este es un separador de tabulación que nos permite decir a partir de qué posición queremos que nos salga la impresión.

100 PRINT TAB(10) "LUIS" TAB(20) "JAVIER" TAB(30) "JOSE"

	LUIS	JAVIER	JOSE
Posición	10	20	30

Ejemplos:

```
10 PRINT 1,2,3
20 PRINT 1;2;3
30 END

1        2        3
123
```

```
10 LET A$="LUIS"
20 LET B$="ANDRES"
30 LET C$=" "
40 PRINT A$;C$;B$
50 PRINT A$;B$
60 PRINT A$,B$
70 PRINT TAB(5)A$ TAB(15)B$
```

SIGUE

```
LUIS  ANDRES
LUISANDRES
LUIS     ANDRES
    LUIS          ANDRES
```

READ — DATA — RESTORE

Las sentencias READ y DATA se utilizan conjuntamente para asignar valores a variables.

El formato de la sentencia READ es el siguiente:

(n.º de línea) READ V_1, V_2, V_3 ....., V_n

donde V_1, V_2 ..., V_n representan variables a las que hay que asignar un valor.

Los valores que queremos asignar a las variables de la sentencia READ, se los daremos por medio de una sentencia DATA, que tiene el formato:

(n.º de línea) DATA X_1, X_2 ..., X_n

donde X_1X_2... X_n son los valores asignados a V_1, V_2..., V_n

Estos valores, al igual que las variables en READ, están separados por comas y se escribirán en el mismo orden que las variables a las que van a asignarse.

Ejemplos:

```
10 READ A,B,C,D,E,F,G,H,I,J
20 DATA 1,2,3,4,5,6,7,8,9,10
```

SIGUE

```
30 PRINT A;B;C;D;E;F;G;H;I;J
40 END

12345678910
```

Se asignan los valores de 1 a 10 a 10 variables.

```
10 DATA 1,2,3,4
20 READ A,B,C,D,E,F,G,H,I,J
30 DATA 5,6,7
40 DATA 8,9,10
50 PRINT A;B;C;D;E;F;G;H;I;J
60 END

12345678910
```

Las sentencias 10, 20, 30, 40 tienen el mismo efecto que las sentencias 10 y 20 del ejemplo anterior.

Una sentencia DATA en un programa no necesita corresponderse con una sentencia READ específica. Esto es debido a que se crea un archivo de datos antes de la ejecución del programa (el archivo **interno).** Contiene todos los valores de las sentencias Data en el programa en el orden de su aparición por número de línea.

Cada vez que se ejecuta una sentencia Read le asigna los valores necesarios y se queda apuntando al siguiente valor que tiene que asignar.

La sentencia RESTORE hace que el siguiente valor a asignar vuelva a ser el primero, es decir, restaura el apuntador al

siguiente valor a asignar. Con lo cual podemos volver a utilizar las listas de valores de todas las sentencias DATA anteriores al RESTORE.

Más ejemplos:

```
10 READ A,B,C
20 DATA 1,2,3,4,5,6,7,8,9,10
30 PRINT A;B;C
40 READ D,E,F,G
50 PRINT D,E,F,G
60 END

123
4          5          6
7
```

No es preciso leer todos los valores de archivo de datos a la vez.

```
10 READ A,B,C,D,E
20 DATA 1,2,3,4
30 END

OUT OF DATA
```

Dará error si tratamos de leer más valores de los que hay en el archivo de datos.

```
10 READ A,B,C
20 DATA 15,25,35,5,6,12
30 PRINT A;B;C
40 RESTORE
```

SIGUE

```
50 READ X,Y,Z
60 PRINT X,Y,Z
70 END

152535
15        25        35
```

Vemos como el RESTORE nos permite volver a asignar los valores desde el principio.

```
 10 DATA "Mayor",45,3
 30 READ DOMICILIO$,NUMERO,PISO
 40 RESTORE
 60 READ DOMICIL1$,NUMER1,PIS1
 70 DATA "Luis",24,1
 80 DATA "Jose",30,2
 90 READ NOMBRE$,EDAD,PUERTA
100 READ NOMBR1$,EDA1,PUERT1
```

Vemos como utilizando el RESTORE podemos utilizar los datos comunes para Luis y José que son vecinos y viven en el mismo piso.

VARIABLE	VALOR
NOMBRE$	LUIS
EDAD	24
DOMICILIO$	MAYOR
NUMERO	45
PISO	3
PUERTA	1
NOMBR1$	JOSE
EDA1	30
DOMICILI1$	MAYOR
NUMER1	45
PIS1	3
PUERT1	2

3. FUNCIONES



Hay una serie de funciones que se utilizan con mucha frecuencia. Sería, pues de gran utilidad tenerlas definidas siempre

para poder usarlas cuando conviniera. EL BASIC nos da la facilidad de tener algunas ya predefinidas para facilitar nuestra tarea de programas.

Veámos las más elementales.

ABS (X)

Nos devuelve el valor absoluto de X

PRINT ABS (7*(—5))
35

ATN (X)

Nos devuelve el valor del arco (en radianes) cuya tangente es X

PRINT ATN (3)
1.24905

COS (X)

Nos devuelve el coseno de X, siendo X un ángulo expresado en radianes.

PRINT 2 * COS (.4)
1.84212

EXP (X)

Nos devuelve el valor e^x (e = 2.718282)

PRINT EXP (4)
54.5981

INT (X)

Nos devuelve el valor del mayor número entero que sea igual o menor que X.

PRINT INT (3.24)
3
PRINT INT (—3.24)
—4

LOG (X)

Nos devuelve el logaritmo en base e de X.

PRINT LOG (45/7)
1.86075

RND (X)

Nos devuelve un número al azar de valor comprendido entre 0 y 1. Es posible conseguir que estos números aleatorios estén comprendidos en otro margen cualquiera usando la fórmula:

(B—A) * RND (0) + A

Así los números aleatorios generados serán $A < n < B$. Normalmente la X no tiene significado, aunque hay ordenadores en los que si queremos obtener números aleatorios entre 1 y N, podremos poner RND (N).

SGN (X)

Nos devuelve +1 si X es positivo, —1 si es negativo y 0 si el valor de X es nulo.

```
PRINT SGN (—7)
—1
PRINT SGN (0)
0
PRINT SGN (7)
1
```

SIN (X)

Nos devuelve el seno del ángulo (X) que debe de estar expresado en radianes.

```
PRINT SIN (1.5)
     .997495
```

SQR (X)

Nos devuelve la raíz cuadrada del argumento (X).

```
PRINT SQR (10)
     3.16228
```

TAN (X)

Nos devuelve la tangente del ángulo (X) que debe de estar expresado en radianes.

DEF FNK (X)

En algunos programas podemos tener fórmulas matemáticas que se repitan con cierta frecuencia a lo largo del programa. El BASIC nos permite definir nuestra propia función, que después podemos llamar como si se tratase de una función trigonométrica, de las vistas anteriormente. La función sólo se define una vez al principio del programa.

Para definirla se escribe primero la sentencia **DEF** y después tres letras de las cuales, las dos primeras han de ser siempre **FN**.

n.º de línea DEF FNK (X) = función (X)

Ejemplo:

```
10 DEF FNA(X)=(X+X*(X+1))
20 PRINT FNA(2)
30 END
```

4. CADENAS DE CARACTERES Y VARIABLES DE CADENA

El Basic también permite el manejo de cadenas alfanuméricas, ya sean constantes o variables. Recordemos que las constantes han de ir cerradas entre comillas y las variables tienen que llevar un símbolo $ detrás del nombre de variable.

Los espacios en blanco se consideran en Basic como caracteres. Así la cadena:

"LUIS MANUEL"

será diferente de la cadena

"LUISMANUEL"

Con las cadenas alfanuméricas se puede efectuar la operación de concatenación (+).

Ejemplo:

```
 10 LET A$="Juan"
 20 LET B$=" "
 30 LET C$="Jose"
 40 LET D$=A$+B$+C$
 50 LET E$=A$+C$
 60 LET F$=A$+" "+"ALFONSO"
 70 PRINT D$
 80 PRINT E$
 90 PRINT F$
100 END

 Juan Jose
 JuanJose
 Juan ALFONSO
```

Una cadena que contenga números como ‘‘795’’, no tiene sentido numérico y no se debe de confundir con la constante numérica 795.

La longitud máxima de una cadena es 255 caracteres.

RESUMEN

REPASO CAPITULO 4º

P1) Decir **qué imprime** el siguiente programa:

```
10 REM DIFERENCIAS DE IMPRESION
20 LET A=50
30 PRINT A
40 PRINT "A"
50 END
```

P2) **Qué salida hará** por pantalla el siguiente programa:

```
10 PRINT 3+2
20 PRINT "3 + 2"
30 END
```

P3) Hacer un programa que me **imprima la suma y la resta** de dos números entrados por teclado.

P4) Hacer un programa que, entrándole **el radio**, nos **calcule** la longitud de la **circunferencia** y nos la **imprima**.

P5) Hacer un programa que, entrando la **base y la altura** de un triángulo rectángulo, **nos dé la base**, la **altura** y el **área**.

P6) Hacer un programa que, entrando 3 números, nos diga su **suma** y su **media aritmética**.

P7) Hacer un programa que, dándole los grados, **nos dé seno, coseno** y **tangente**.

P8) Hacer un programa que, dado un número positivo, **nos dé el logaritmo decimal**.

P9) **Qué valor se imprimirá** cuando se ejecute el siguiente programa:

```
10 LET N$="AGOSTO"
20 LET D$="LUNES"
30 LET E$="DIEZ"
40 LET B$=" "
50 LET FE$=D$+B$+E$+B$+"de"+B$+N$
60 PRINT "fecha:";B$;FE$
```

P10) Hacer un programa que, **dándole la distancia en Km.** y **el tiempo en horas** empleado, nos **calcule la velocidad.**

P11) Hacer un programa que dándole **los catetos** de un triángulo restángulo nos **calcule la hipotenusa.**

STOP

No pasar la página hasta haber realizado los ejercicios planteados.

SOLUCIONES



R1)

```
50
A
```

R2)

```
5
3 + 2
```

R3)

```
10 REM SUMA y RESTA
20 INPUT "Teclea DOS numeros: ";A,B
30 LET S=A+B
40 LET R=A-B
50 PRINT "La suma es= ";S
60 PRINT "La resta es=";R
70 END
```

R4)

```
10 REM LONGITUD DE LA CIRCUNFERENCIA
20 INPUT "Teclea el radio: ";R
30 LET PI=3.14159
40 LET L=2*PI*R
50 PRINT "La longitud de la circunferencia mide: ";L
60 END
```

R5)

```
10 REM AREA DE UN TRIANGULO
20 INPUT "Cuanto mide la Base?.... ";B
30 INPUT "Cuanto mide la Altura?.. ";A
40 LET AREA=B*A/2
```

SIGUE

```
50 PRINT "BASE=   ";B
60 PRINT "ALTURA=";A
70 PRINT "AREA=   ";AREA
80 END
```

R6)

```
10 REM SUMA Y MEDIA ARITMETICA
20 INPUT "Teclea TRES numeros: ";A,B,C
30 LET S=A+B+C
40 M=S/3
50 PRINT "La SUMA es = ";S
60 PRINT "La MEDIA ES= ";M
70 END
```

R7) Para pasar los grados a radianes

Angulo * 3.14159 / 180

```
10 REM CALCULO SENO COSENO TANGENTE
20 INPUT "Grados del ANGULO a convertir?...";ANG
30 LET S=SIN(ANG*3.14159/180)
40 LET C=COS(ANG*3.14159/180)
50 LET T=TAN(ANG*3.14159/180)
60 PRINT "SENO=         ";S
70 PRINT "COSENO=       ";C
80 PRINT "TANGENTE=     ";T
90 END
```

R8) Para calcular un logaritmo en base decimal

SIGUE

```
10 REM LOGARITMO DECIMAL
20 INPUT "Teclea un numero positivo: ";N
```

```
30 LET R=LOG(N)/LOG(10)
40 PRINT "El log.decimal de: ";N;"Es igual a ";R
50 END
```

R9)

```
fecha: LUNES DIEZ de AGOSTO
```

R10)

```
10 REM CALCULO DE VELOCIDAD
20 INPUT "Distancia en Kilometros?... ";K
30 INPUT "Tiempo en Horas?........... ";H
40 LET V=K/H
50 PRINT "La VELOCIDAD es de ";V;" Km/Hora"
60 END
```

R11)

```
10 REM CALCULO DE LA HIPOTENUSA
20 INPUT "Teclea la medida de los CATETOS: ";A,B
30 LET H=SQR(A^2+B^2)
40 PRINT "La HIPOTENUSA MIDE:             ";H
50 END
```


5

INDICE DE CAPITULO

CAPITULO 5º

BUCLES Y TRANSFERENCIAS DE CONTROL



1. FOR ... NEXT

A menudo necesitamos que el ordenador repita una serie de intrucciones muchas veces para distintos valores de una variable que se va incrementando cada vez en un valor determinado. Para conseguirlo nos son de gran utilidad los bucles FOR ... NEXT.

Estas sentencias se encargan de que se recorra el cuerpo del bucle las veces indicadas, mediante una condición.

Veámos el formato de una instrucción FOR ... NEXT:

n.º línea FOR X = V1 TO V2 STEP V3
n.º línea NEXT X

Donde X es el nombre de la variable, y V1 V2 V3 son cantidades numéricas, que pueden ser constantes, expresiones o variables.

V1 — es el valor inicial que toma X

V2 — es el valor final que toma X

V3 — es el incremento que tendrá X

Es decir, entre FOR y NEXT estará el cuerpo del bucle que no es ni más ni menos que un conjunto de instrucciones.

Cuando se ejecuta el FOR X toma el valor V1 y ejecuta el cuerpo del bucle con ese valor, al llegar al NEXT, el valor de X se incrementa en V3 y vuelve a ejecutar el cuerpo del bucle con ese nuevo valor, esta operación se repite hasta que al ejecutar el NEXT el valor X sea igual o mayor que V2.

Si se omite el valor V3 se supone que el incremento es 1.

Ejemplo:

```
10 INPUT "Dime tu nombre:........... ";A$
20 INPUT "Numero de repeticiones?... ";N
30 FOR I=1 TO N
40 PRINT A$
50 NEXT I
60 END
```

SIGUE

DIME TU NOMBRE LUIS MANUEL [RETURN]
NUMERO DE REPETICIONES 5 [RETURN]

```
LUIS MANUEL
LUIS MANUEL
.
.
LUIS MANUEL
```

Para imprimir los cuadrados de los 5 primeros números:

```
10 FOR I=1 TO 5
20 PRINT I^2,I
30 NEXT I
40 END

1        1
4        2
9        3
16       4
25       5
```

Para imprimir los números pares entre 20 y 30:

```
10 REM PARES DEL 20 al 30
20 FOR I=20 TO 30 STEP 2
30 PRINT I
40 NEXT I
50 END

22
24
26
28
30
```

Utilizando cadenas:

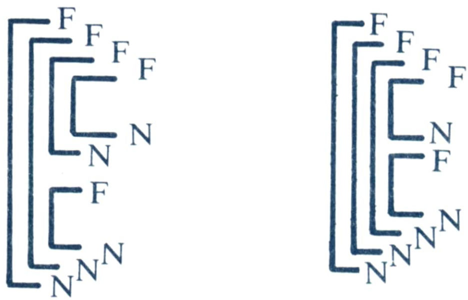
```
10 DATA "H","o","l","a"," ","a","m","i","g","o"," ",
        "q","u","e"," ","t","a","l","?"
20 FOR I=1 TO 19
30 READ A$
40 LET C$=C$+A$
50 NEXT I
60 PRINT C$
70 END

Hola amigo que tal?
```

2. BUCLES ANIDADOS

Dentro de un For ... next, podemos encontrarnos **otro** For ... next; éste es el caso de los bucles anidados. Pero debemos tener en cuenta que el primer FOR que se encuentra en sentido descendente ha de corresponder al último NEXT encontrado, y el último NEXT debe corresponder al primer FOR.

Es decir, han de tener la estructura correspondiente a los siguientes gráficos:



Correctos



Incorrectos

Ejemplos:

```
10 REM EJEMPLOS
20 FOR I=1 TO 4
30 FOR J=10 TO 40 STEP 10
40 PRINT I" ";J" ";I*J
50 NEXT J
60 NEXT I
70 END
```

SIGUE

```
1 10 11
1 20 21
1 30 31
1 40 41
2 10 12
2 20 22
2 30 32
2 40 42
3 10 13
3 20 23
3 30 33
3 40 43
4 10 14
4 20 24
4 30 34
4 40 44
```

Veámos el siguiente ejemplo:

Queremos evaluar la fórmula

```
C=(1+R)^T
```

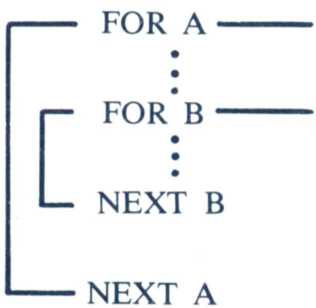
R puede valer = 0.01, 0.05, 0.09

T puede valer = 1 y 3

Es decir,

T = 1 y R = 0.01, 0.05, 0.09
T = 3 y R = 0.01, 0.05, 0.09

```
10 REM EVALUAR FORMULA
20 FOR T=1 TO 3 STEP 2
30 FOR R=0.01 TO 0.09 STEP 0.04
40 LET C=(1+R)^T
50 PRINT T,R,C
60 NEXT R
70 NEXT T
80 END
```



• Para cada valor de la variable del ciclo exterior (A), la variable del ciclo interior (B) recorre todos los valores. Es decir, el ciclo interior se realiza por completo.

Cuando el ciclo interior se ha completado, el control pasa a las instrucciones siguientes en el ciclo exterior. El proceso continúa hasta completar el ciclo exterior.

• Puede haber varios ciclos anidados.

• Cada ciclo ha de tener una variable contador diferente.

• Los ciclos no pueden solaparse.

```
┌FOR A ────
├FOR B ────
├NEXT A
└NEXT B
```

no es válido

Estudiar y analizar el siguiente ejemplo:

```
10 FOR X=2 TO 9
20 PRINT TAB(X-1);
30 FOR A=1 TO 8
40 PRINT "# ";
50 NEXT A
60 PRINT
70 NEXT X
80 END
```

```
# # # # # # # #
 # # # # # # # #
  # # # # # # # #
   # # # # # # # #
    # # # # # # # #
     # # # # # # # #
      # # # # # # # #
       # # # # # # # #
```

3. BIFURCACION INCONDICIONAL — GOTO



Las instrucciones en el lenguaje Basic se ejecutan normalmente según el orden creciente de sus números de línea. Pero a veces es necesario alterar este orden normal de ejecución y **"saltar"** a una línea posterior o anterior. Este es el papel de la intrucción GOTO.

Su formato es:

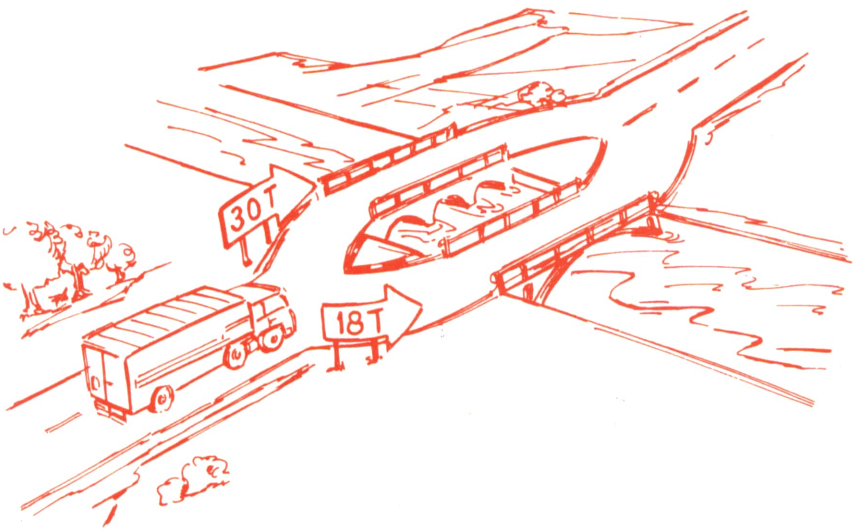
(n.º de línea) GOTO n.º línea

Cuando se ejecuta una instrucción GOTO, en lugar de ejecutar luego la siguiente instrucción, salta a la línea cuyo número aparece a la derecha de la sentencia GOTO, y la ejecuta.

```
10 REM EJEMPLO
20 PRINT "Hola !!!"
30 GOTO 20
40 END
```

Este programa no se acabará nunca, pues cuando llega a la instrucción 30 vuelve a la 20 y nunca ejecutará la 40.

4. IF ... BIFURCACION CONDICIONAL



— IF ... GOTO

Por medio de esta sentencia, el programa comprueba una condición y, dependiendo de su veracidad, transfiere el control a la línea de programa indicada.

n.º línea IF condición GOTO n.º línea

Esta condición suele ser una comparación mediante operadores de relación vistos en capítulos anteriores (<, >, <>, =, =<, =>).

Se pueden comparar variables, constantes y expresiones, ya sean numéricas o alfanuméricas.

```
10 REM EJEMPLO
20 INPUT "Teclear un NUMERO: ";N
30 IF N>=0 GOTO 60
40 PRINT "El NUMERO es negativo"
50 GOTO 70
60 PRINT "El NUMERO es positivo"
70 END
```

Otro formato:

— IF ... THEN

(n.º línea IF condición THEN (n.º línea) o Instrucción

Su utilización es similar a la anterior. Es decir, si se cumple la condición, puede transferir el control al n.º de línea especificada o realizar la instrucción indicada.

Veámos algunos ejemplos:

```
10 REM SUMAR NUMEROS
20 INPUT "Dime un numero (del 1 al 100) ?... ";A
30 IF A>100 OR A<0 GOTO 20
40 INPUT "Dime un numero (del 1 al 100) ?... ";B
```

SIGUE

```
 50 IF B>100 OR B<0 GOTO 40
 60 LET C=A+B
 70 PRINT TAB(13);A
 80 PRINT TAB(11);"+ ";B
 90 PRINT TAB(12);"_____"
100 PRINT TAB(13);C
```

DIME UN NUMERO (1—100)? 56 [RETURN]

DIME UN NUMERO (1—100)? 75 [RETURN]

```
  56
+ 75
_____
 131
```

Las instrucciones 30 y 50 nos sirven para comprobar que los números a sumar están comprendidos entre 1 y 100.

```
 10 REM PROGRAMA PARA COMPARAR DOS NUMEROS
 20 INPUT "Introduce DOS numeros: ";A,B
 30 IF A>B GOTO 70
 40 IF A<B GOTO 90
 50 REM ACABA LA EJECUCION CUANDO A=B
 60 GOTO 110
 70 PRINT A;" es MAYOR que ";B
 80 GOTO 20
 90 PRINT A;" es MENOR que ";B
100 GOTO 20
110 PRINT "Finalizado el trabajo"
120 END
```

SIGUE

```
Introduce DOS numeros: 10, 12
10 es MENOR que 12
Introduce DOS numeros: 15, 8
15 es MAYOR que 8
Introduce DOS numeros: 10, 10
Finalizado el trabajo
```

5. GOSUB Y RETURN

Una subrutina es una secuencia de instrucciones que realizan una operación o trabajo que se utiliza en más de un punto del programa.

A estas subrutinas, nomalmente, se las sitúa al final del programa, antes de la sentencia END. El manejo de las mismas se lleva a cabo con las instrucciones GOSUB y RETURN.

Su formato es:

n.º línea GOSUB n.º de línea

Cuando el programa se encuentra con esta sentencia transfiere el control o bifurca al número de línea especificado y empieza a ejecutar las instrucciones de la subrutina, hasta que se encuentra con una instrucción RETURN, que hace que vuelva a la línea siguiente a aquella en la que se encontró el

GOSUB. De este modo el programa continuará su proceso en el mismo punto donde había interrumpido su ejecución.

Veámos un programa que utiliza GOSUB y RETURN

```
 10 REM PROGRAMA SUBRUTINA
 20 DATA 4,7,12,22,-1
 30 DATA 48,13,125,-1
 40 DATA 125,13,63,45,46,83,-1
 50 GOSUB 200
 60 M1=M
 70 GOSUB 200
 80 M2=M
 90 GOSUB 200
100 M3=M
110 PRINT "La MEDIA de la primera serie= ";M1
120 PRINT "La MEDIA de la segunda serie= ";M2
130 PRINT "La MEDIA de la tercera serie= ";M3
140 END
  .
  .
  .
200 REM SUBRUTINA
210 N=0
220 S=0
230 READ X
240 IF X<0 THEN 280
250 S=S+X
260 N=N+1
270 GOTO 230
280 M=S/N
290 RETURN
```

RESUMEN

REPASO CAPITULO 5º

P1) Hacer un programa que escriba 100 veces vuestro nombre.

P2) Hacer un programa para calcular las raíces de una ecuación de 2º grado.

P3) Hacer un programa que nos dé la tabla de multiplicar del 9, de la forma:

9 * ... = ...

P4) Hacer un programa que nos calcule el siguiente número:

$$a = 1 + 1/2 + 1/3 + 1/4 + \ldots + 1/100$$

P5) Lo mismo para:

$$b = 1/2 + 1/4 + 1/6 + 1/8 + \ldots + 1/100$$

P6) Lo mismo para:

$$c = 1/2 + 1/4 + 1/8 + 1/16 + \ldots + 1/512$$

P7) Hacer un programa que le introduzcamos 20 números y nos dé su suma y su media aritmética.

P8) Hacer un programa que introduciéndose 20 números positivos, nos diga cual es el mayor y su número de orden respecto a la entrada.

STOP

No pasar la página hasta haber realizado los ejercicios planteados.

SOLUCIONES

R1)

```
10 REM PROGRAMA NOMBRE
20 INPUT "Dime tu nombre?... ";N$
30 FOR I=1 TO 100
40 PRINT N$
50 NEXT I
60 END
```

R2)

```
10 REM ECUACIONES DE SEGUNDO GRADO
20 INPUT "Valor de A ";A
30 INPUT "Valor de B ";B
40 INPUT "Valor de C ";C
50 LET X1=(-B+(SQR(B^2-A*C))/(2*A))
60 LET X2=(-B-(SQR(B^2-A*C))/(2*A))
70 IF (B^2-4*A*C)<0 THEN 40
80 PRINT "X1= ";X1
90 PRINT "X2= ";X2
100 END
```

R3)

```
10 REM TABLA DE MULTIPLICAR
20 FOR I=1 TO 9
30 PRINT "9 x ";I " = ";9*I
40 NEXT I
50 END
```

R4)

```
10 REM SERIE A
20 FOR I=1 TO 100
30 LET S=S+1/I
40 NEXT I
50 PRINT "El numero es= ";S
```

R5)

```
10 REM SERIE B
20 FOR I=2 TO 100 STEP 2
30 LET S=S+1/I
40 NEXT I
50 PRINT "El numero es: ";S
```

R6)

```
10 REM SERIE C
20 FOR I=1 TO 9
30 LET S=S+1/2^I
40 NEXT I
50 PRINT "El numero es: ";S
```

R7)

```
10 REM SUMA Y MEDIA
20 FOR I=1 TO 20
30 INPUT "ENTRAR NUMERO: ";N
40 LET S=S+N
50 NEXT I
60 PRINT "La suma de los 20 numeros es: ";S
70 PRINT "La media aritmetica es:        ";S/20
80 END
```

R8)

```
10 REM BUSQUEDA DEL MAYOR
20 FOR I=1 TO 20
```

SIGUE

```
30 INPUT "ENTRAR NUMERO: ";N
40 IF N>C THEN LET C=N:LET K=I
50 NEXT I
60 PRINT "El MAYOR es:    ";C
70 PRINT "Ha sido introducido en la ";K;" posicion"
80 END
```


6

INDICE DE CAPITULO

CAPITULO 6º

LISTAS Y TABLAS



1. VARIABLES DIMENSIONADAS

Hasta el momento, hemos visto variables que contienen un solo valor y tienen un nombre de variable. El Basic nos permite utilizar otro tipo de variables, formadas por un nombre de variable y un número natural entre paréntesis.

Una lista o tabla es un conjunto de elementos dispuestos en forma adecuada y que guardan una relación concreta entre sí.

Antes de utilizar una lista, debemos decirle al ordenador de qué tipo y qué longitud va a tener, para que éste prevea el espacio que va a necesitar en la memoria. Esto se llama Dimensionar y se utiliza la instrucción:

nº línea DIM nombre de variable (nº de elementos)

Por ejemplo:

10 DIM M$(12)

Esta instrucción me reservaría **13** variables alfanuméricas que las podría distinguir por sus subíndices y serían: (M$(0), M$(1), M$(2) ... M$(12)).

NOTA:

Si no necesito que me reserve el M$(0), pues prefiero que el subíndice empiece por el valor 1, algunos ordenadores utilizan la instrucción: **nº linea OPTION BASE 1.**

En nuestro ordenador esto no es posible.

Ejemplo:

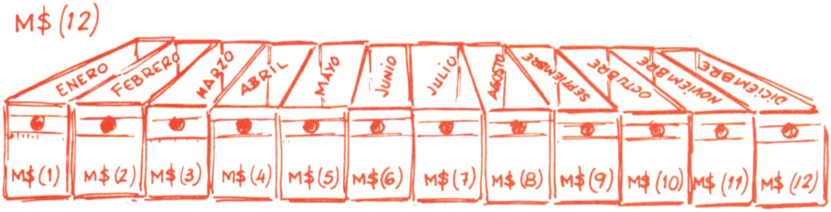
```
10 DIM M$(12)
30 DATA ENERO,FEBRERO,MARZO,ABRIL,MAYO
40 DATA JUNIO,JULIO,AGOSTO,SEPTIEMBRE
```

SIGUE

```
50 DATA OCTUBRE,NOVIEMBRE,DICIEMBRE
60 FOR I=1 TO 12
70 READ M$(I)
80 NEXT I
 .
 .
 .
```

Esta parte de programa nos ha creado una lista en Memoria en la que he guardado los meses del año, de la forma siguiente:

```
M$(1)=ENERO
M$(2)=FEBRERO
   .
   .
   .
M$(12)=DICIEMBRE
```



Siguiendo con el programa anterior:

```
 90 INPUT "Numeros de mes: <?> , Fin: <0>";Y
100 IF Y=0 GOTO 140
110 IF Y<0 OR Y>12 GOTO 90
120 PRINT "EL MES ES: ";M$(Y)
130 GOTO 90
140 END
```

Como vemos, este programa nos pasará el número de mes a su nombre escrito.

Un problema que se suele presentar con frecuencia será encontrar el mayor de los números de una lista y el lugar que ocupa en ella. Por ejemplo:

P(1)	P(2)	P(3)	P(4)	P(5)	P(6)	P(7)	P(8)
16	10	9	2	-3	-1	23	4

el mayor es el 23 y la posición que ocupa la 7.

El tamaño de la lista del ejemplo hace que no reparemos en todo el proceso que supone la búsqueda de valor más alto. Pero si nos imaginamos una lista de 1.000 elementos, necesitaremos recorrernos la lista de principio a fin. Utilizaremos una variable M para guardar el mayor en cada momento y una variable P para recordar la posición del último mayor encontrado.

Veámos:

```
10 REM BUSQUEDA DEL MAYOR
30 DIM P(100)
40 FOR I=1 TO 100
50 LET P(I)=RND(I)*100
60 NEXT I
```

Generación de la lista por n.os aleatorios

SIGUE

```
100 LET M=P(1)
110 LET P=1
120 FOR I=2 TO 100
130 IF M>P(I) GOTO 160
140 LET M=P(I)
150 LET P=I
160 NEXT I
170 PRINT "El numero MAYOR de la serie es el: ";M
180 PRINT "Ocupa la posicion:                 ";P
190 END
```

Otro problema que se presenta bastante a menudo, tratando con listas, es tener que ordenarlas de menor a mayor o viceversa.

A continuación explicamos un método sencillo para la ordenación de una lista, evidentemente existen otros muchos, pero aquí no tienen cabida.

Este procedimiento consiste en:

Comparar un elemento con el siguiente; si están en orden se dejan como están y si no, se intercambian sus posiciones y así con todos los elementos de la lista.

Este proceso se repetirá hasta que en un recorrido completo de la lista no se efectúe ningún intercambio.

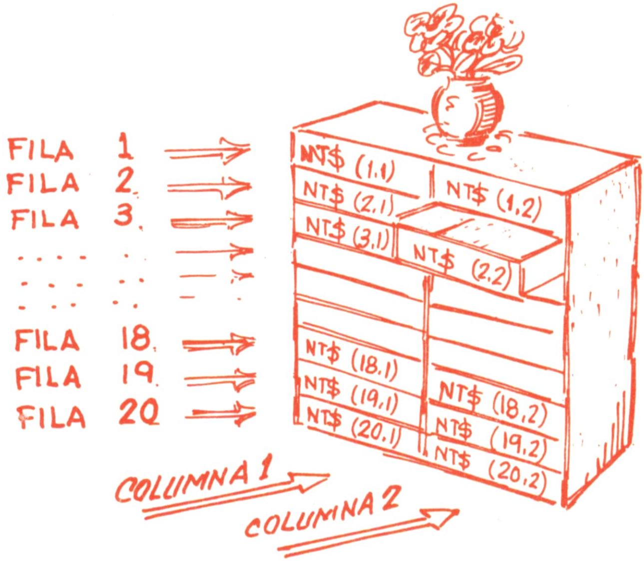
```
10 REM ORDENACION DE UNA LISTA
30 DIM P(100)
40 FOR I=1 TO 100
50 LET P(I)=RND(I)*100
60 NEXT I
```

SIGUE

```
70 REM ORDENACION
80 HOME:VTAB 12:PRINT "!!! ESTOY ORDENANDO LA SERIE!!!"

100 LET X=0
110 FOR I=1 TO 99
120 IF P(I)<=P(I+1) GOTO 180
130 REM INTERCAMBIO
140 LET M=P(I)
150 LET P(I)=P(I+1)
160 LET P(I+1)=M
170 LET X=1
180 NEXT I
190 IF X=1 GOTO 100
200 FOR I=1 TO 100
210 PRINT P(I);
220 NEXT I
230 END
```

2. VARIABLES DE DOS DIMENSIONES (TABLAS)



Hasta ahora, sólo hemos utilizado variables dimensionadas de un sólo subíndice. Veámos ahora las de **dos subíndices**. Por ejemplo:

P(3,4)

Estas variables se utilizan para representar las tablas.

		columnas	
filas	— P(1,1)	P(1,2)	P(1,3)
	— P(2,1)	P(2,2)	P(2,3)
	— P(3,1)	P(3,2)	P(3,3)

Esta sería una tabla de 3 filas por 3 columnas y para utilizarla en nuestro programa hubiésemos necesitado poner:

DIM P(3,3)

con lo cual, reservamos espacio para 9 variables que luego podremos utilizar refiriéndonos a ellas con sus nombres y sus subíndices (fila, columna).

Evidentemente, las variables pueden ser numéricas o alfanuméricas T$(10,7).

Para mostrar el manejo de las variables con dos índices vamos a ver un programa muy simple:

Se trata de asignar mediante INPUT, valores a la variable A(I,J) de 12 filas y 3 columnas; y posteriormente, imprimir la tabla en la pantalla del ordenador.

```
10 REM IMPRESION DE UNA TABLA
30 DIM A$(12,3)
40 FOR I=1 TO 12
50 FOR J=1 TO 3
60 INPUT A(I,J)
70 NEXT J
```

SIGUE

```
 80 NEXT I
 90 FOR I=1 TO 12
100 FOR J=1 TO 3
110 PRINT A(I,J);
120 NEXT J
130 PRINT
140 NEXT I
150 END
```

Se observa el (;) al final de línea 110, que es necesario para que las impresiones de los 3 elementos de cada línea de la tabla se efectúen en la misma línea de pantalla. La instrucción PRINT de la línea 130, es necesaria para que la impresión del primer elemento de cada línea de la tabla se haga al comienzo de una línea de pantalla.

RESUMEN

REPASO CAPITULO 6º

P1) Hacer un programa que **introduzca** valores de un vector de **10 posiciones** y **obtenga su suma.**

P2) Hacer un programa para **hallar el menor elemento** de un **vector** de 100 posiciones.

P3) Hacer un programa para **hallar el menor elemento** de una **matriz** de 5 × 5.

P4) Hacer un programa que nos **imprima una matriz de 5 × 5,** en la cual el **valor de sus elementos sea la suma de los subíndices que indican su posición.**

P5) Hacer un programa que nos **imprima la tabla de multiplicar del 1 al 9.**

1 2 3 4 5 6 7 8 9
2 4 6 8 10 .....
3 6 9 ...
4 8
5 .
6 :

SIGUE

7
8
9

P6) Hacer un programa que **nos ordene un vector de 100 posiciones de mayor a menor.**

P7) Hacer un programa para **comprobar la siguiente tabla cuadrada** e **imprimir el resultado.**

```
x B B B B
A x B B B
A A x B B
A A A x B
A A A A x
```

Puede ser de cualquier tamaño hasta (14,14)

P8) Hacer un programa que nos **imprima la matriz unidad de 5 × 5.**

P9) Hacer un programa que **entrándole dos matrices A(3 × 3) y B(3 × 3) nos dé como resultado la impresión de una matriz C(3 × 3);** suma de las otras dos.

STOP

No pasar la página hasta haber realizado los ejercicios planteados.

SOLUCIONES

R1)

```
 10 REM SUMA DE UN VECTOR
 30 DIM A(10)
 40 FOR I=1 TO 10
 50 LET A(I)=RND(I)*10
 60 NEXT I
 70 FOR I=1 TO 10
 80 LET S=S+A(I)
 90 NEXT I
100 PRINT "La SUMA del vector es: ";S
110 END
```

R2)

```
 10 REM BUSQUEDA DEL MENOR
 30 DIM P(100)
 40 FOR I=1 TO 100
 50 LET P(I)=RND(I)*100
 60 NEXT I
 70 LET M=P(1)
 80 LET P=1
 90 FOR I=2 TO 100
100 IF M<=P(I) GOTO 130
110 LET M=P(I)
120 LET P=I
130 NEXT I
140 PRINT "El MENOR de la serie es el: ";M
150 PRINT "Ocupa el lugar:                ";P
160 END
```

R3)

```
10 REM MENOR DE MATRIZ 5 x 5
30 DIM A(5,5)
40 FOR I=1 TO 5
50 FOR J=1 TO 5
60 LET A (I,J)=RND(1)*25
70 NEXT J
80 NEXT I
90 LET M=A(1,1)
100 FOR I=1 TO 5
110 FOR J=1 TO 5
120 IF M>A(I,J) THEN LET M=A(I,J)
130 NEXT J
140 NEXT I
150 PRINT "El MENOR es: ";M
160 END
```

R4)

```
10 REM SUMA DE SUBINDICES
30 DIM A(5,5)
40 FOR I=1 TO 5
50 FOR J=1 TO 5
60 LET A(I,J)=I+J
70 PRINT A(I,J);" ";
80 NEXT J
90 PRINT
100 NEXT I
110 END
```

R5)

```
10 REM TABLA de MULTIPLICAR
30 DIM A(9,9)
40 FOR I=1 TO 9
50 FOR J=1 TO 9
```

SIGUE

```
 60 LET A(I,J)=I*J
 70 NEXT J
 80 NEXT I
 90 FOR I=1 TO 9
100 FOR J=1 TO 9
110 PRINT A(I,J)"   ";
120 NEXT J
130 PRINT
140 NEXT I
150 END
```

R6)

```
 10 REM ORDENACION DE UNA LISTA
 30 DIM P(100)
 40 FOR I=1 TO 100
 50 LET P(I)=RND(I)*100
 60 NEXT I
 70 REM ORDENACION
 80 LET X=0
 90 FOR I=1 TO 99
100 IF P(I)>=P(I+1) THEN 150
110 LET M=P(I)
120 LET P(I)=P(I+1)
130 LET P(I+1)=M
140 LET X=1
150 NEXT I
160 IF X=1 TO 80
170 FOR I=1 TO 100
180 PRINT P(I);
190 NEXT I
200 END
```

R7)

```
 10 REM TABLA CUADRADA
 20 DIM A$(14,14)
 30 INPUT "TAMAÑO DESEADO ?... ",N
 40 FOR I=1 TO N
 50 FOR J=1 TO N
 60 IF I<>J THEN 80
 70 LET A$(I,J)="X":GOTO 110
 80 IF I>J THEN 100
 90 LET A$(I,J)="B":GOTO 110
100 LET A$(I,J)="A"
110 NEXT J
120 NEXT I
130 FOR I=1 TO N
140 FOR J=1 TO N
150 PRINT " ";A$(I,J);
160 NEXT J
170 PRINT
180 NEXT I
190 END
```

R8)

```
 10 REM MATRIZ UNIDAD DE 5 x 5
 30 DIM A(5,5)
 40 FOR I=1 TO 5
 50 FOR J=1 TO 5
 60 LET A(I,J)=0
 70 IF I=J THEN LET A(I,J)=1
 80 NEXT J
 90 NEXT I
100 FOR I=1 TO 5
110 FOR J=1 TO 5
120 PRINT A(I,J);
130 NEXT J
```

SIGUE

```
140 PRINT
150 NEXT I
160 END
```

R9)

```
 10 REM SUMA DE DOS MATRICES
 30 DIM A(3,3),B(3,3),C(3,3)
 40 FOR I=1 TO 3
 50 FOR J=1 TO 3
 60 INPUT "ELEMENTO DE A "; A(I,J)
 70 NEXT J
 80 NEXT I
 90 FOR I=1 TO 3
100 FOR J=1 TO 3
110 INPUT "ELEMENTO DE B ";B(I,J)
120 NEXT J
130 NEXT I
140 FOR I=1 TO 3
150 FOR J=1 TO 3
160 LET C(I,J)=A(I,J)+B(I,J)
170 NEXT J
180 NEXT I
190 FOR I=1 TO 3
200 FOR J=1 TO 3
210 PRINT C(I,J),
220 NEXT J
230 PRINT
240 NEXT I
250 END
```

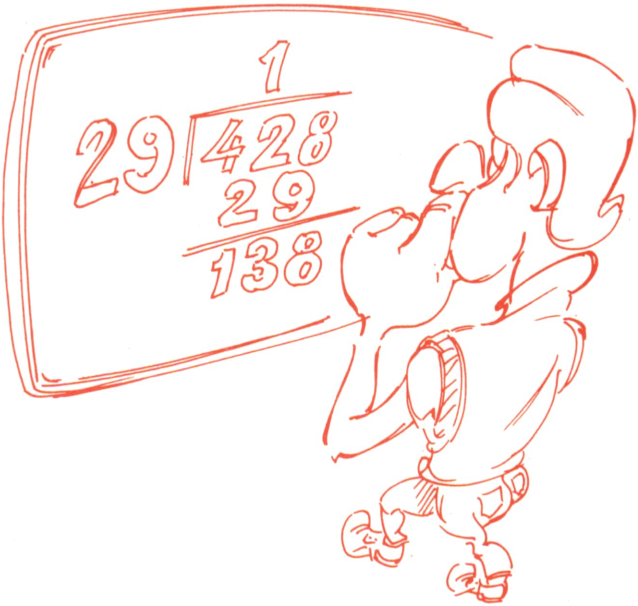
APENDICE

EJERCICIOS DE RESOLUCION DE PROBLEMAS DE MATEMATICAS Y FISICA MEDIANTE EL LENGUAJE BASIC

INDICE

ENUNCIADOS I PARTE

PROBLEMAS DE MATEMATICAS



1. Hacer un programa que nos imprima el **producto cartesiano de los números naturales** del 1 al 9; N × N

   N = (1,2,3,4,5,6,7,8,9)

2. Hacer un programa que dándole los **lados de dos triángulos,** nos diga si son o no **triángulos semejantes.**

3. Hacer un programa que dándole el **radio** nos calcule:
    - **longitud de la circunferencia**
    - **área del círculo**
    - **área de la esfera**
    - **volumen de la esfera**

4. Hacer un programa que nos calcule el **área total** y el **volumen de un cono.**

5. Lo mismo con un **cilindro.**

6. Hacer un programa que nos calcule las soluciones de un **sistema de ecuaciones** de la siguiente forma:

    ax + by = c

    alx + bly = cl

7. Hacer un programa que nos calcule el **tanto por ciento,** dándole la cantidad y el porcentaje.

8. Hacer un programa que me permita calcular el **interés** o el **capital,** o el **rédito,** o el **tiempo en meses,** dándole el resto de los datos.

9. Hacer un programa que dándole los **catetos de un triángulo rectángulo me diga la hipotenusa;** o dándole un **cateto y la hipotenusa,** me dé el **otro cateto.**

10. Hacer un programa, que dándole la **diagonal de un cuadrado** me dé el **área** y los **lados** del mismo.

11. Hacer un programa para resolver **ecuaciones** del tipo

$$ax^2 + bx = 0$$

12. Hacer un programa para resolver **ecuaciones** del tipo

$$ax^2 + c = 0$$

13. Hacer un programa para resolver **ecuaciones** del tipo

$$ax^2 + bx + c = 0$$

14. Hacer un programa que nos calcule la **media aritmética** de una serie de números.

STOP

No pasar la página hasta haber realizado los ejercicios planteados.

SOLUCIONES I PARTE

1.

```
10 REM PRODUCTO CARTESIANO
20 FOR I=1 TO 9
30 FOR J=1 TO 9
40 PRINT I;",";J
50 NEXT J
60 NEXT I
70 END
```

2. Dos triángulos son equivalentes si sus lados son proporcionales.

```
10 REM TRIANGULOS EQUIVALENTES
20 INPUT "LADOS DEL 1 TRIANGULO ";A,B,C
30 INPUT "LADOS DEL 2 TRIANGULO ";D,E,F
40 IF A<B THEN LET P=A:LET A=B:LET B=P
50 IF A<C THEN LET P=A:LET A=C:LET C=P
60 IF B<C THEN LET P=B:LET B=C:LET C=P
70 IF D<E THEN LET P=D:LET D=E:LET E=P
80 IF D<F THEN LET P=D:LET D=F:LET F=P
90 IF E<F THEN LET P=E:LET E=F:LET F=P
100 IF (A/D=B/E) AND (B/E=C/F) THEN 130
110 PRINT "LOS TRIANGULOS NO SON EQUIVALENTES"
120 GOTO 140
130 PRINT "LOS TRIANGULOS SON EQUIVALENTES"
140 END
```

3.

```
10 REM CIRCULO,CIRCUNFERENCIA,ESFERA
20 INPUT "ENTRAR EL RADIO ";R
30 LET PI=3.141592
40 LET L=2*PI*R
```

SIGUE

```
50 PRINT "LA LONGITUD DE LA CIRCUNFERENCIA ES: ";L
60 LET A=PI*R^2
70 PRINT "EL AREA DEL CIRCULO ES:        "; A
80 LET AE=4*A
90 PRINT "EL AREA DE LA ESFERA ES:       ";AE
100 LET V=(4*PI*R^3)/3
110 PRINT "EL VOLUMEN DE LA ESFERA ES: ";V
120 END
```

4.

```
10 REM AREA Y VOLUMEN DE UN CONO
20 INPUT "ENTRAR ALTURA: ";H
30 INPUT "ENTRAR RADIO:  ";R
40 LET G=SQR(H^2+R^2): LET PI=3.141592
50 LET A=PI*R*G+PI*R^2
60 PRINT "EL AREA TOTAL ES: ";A
70 LET V=PI*R^2*H/3
80 PRINT "EL VOLUMEN ES: ";V
90 END
```

5.

```
10 REM AREA Y VOLUMEN DE UN CILINDRO
20 INPUT "ENTRAR RADIO   ";R
30 INPUT "ENTRAR ALTURA ";H
40 LET PI=3.141592
50 LET A=2*PI*R*H+2*PI*R^2
60 PRINT "EL AREA TOTAL DEL CILINDRO ES: ";A
70 LET V=PI*R^2*H
80 PRINT "EL VOLUMEN DEL CILINDRO ES: ";V
90 END
```

6. $ax + by = c$

$a1x + b1y = c1$

SIGUE

$$ax = c - by$$

$$\boxed{x = \frac{c - by}{a}}$$

$$a1\left(\frac{c - by}{a}\right) + b1y = c1$$

$$\frac{a1c - a1by}{a} + b1y = c1$$

$$a1c - a1by + ab1y = ac1$$

$$y(ab1 - a1b) = ac1 - a1c$$

$$\boxed{y = \frac{ac1 - a1c}{ab1 - a1b}}$$

```
10 REM ECUACIONES
20 INPUT "VALORES A,B,C,1 ECUACION ";A,B,C
30 IF A=0 OR B=0 OR C=0 THEN 20
40 INPUT "VALORES A,B,C, 2 ECUACION ";A1,B1,C1
50 IF A1=0 OR B1=0 OR C1=0 THEN 40
60 LET Y=(A*C1-A1*C)/(A*B1-A1*B)
70 LET X=(C-B*Y)/A
80 PRINT "EL VALOR X=";X
90 PRINT "EL VALOR Y=";Y
100 END
```

7.

```
10 REM TANTO POR CIENTO
20 INPUT "ENTRAR CANTIDAD";C
30 INPUT "ENTRAR PORCENTAJE";P
40 LET T=C*P/100
50 PRINT "EL TANTO POR CIENTO ES: ";T
60 END
```

8.

```
10 REM INTERES COMPUESTO
20 PRINT "SI QUIERES SABER EL CAPITAL PULSAR<C>"
30 PRINT "SI QUIERES SABER EL REDITO PULSAR <R>"
40 PRINT "SI QUIERES SABER EL TIEMPO PULSAR <T>"
50 PRINT "SI QUIERES SABER EL INTERES PULSAR<I>"
60 INPUT A$
70 IF A$<>"I" AND A$<>"R" AND A$<>"T" AND A$<>"C" THEN 60
80 IF A$="C" THEN 100
90 INPUT "CAPITAL";C
100 IF A$="R" THEN 120
110 INPUT "REDITO";R
120 IF A$="T" THEN 140
130 INPUT "TIEMPO EN MESES";T
140 IF A$="I" THEN 160
150 INPUT "INTERES";I
160 IF A$="I" THEN PRINT "EL INTERES=";C*R*T/1200
170 IF A$="C" THEN PRINT "EL CAPITAL=";(1200*I)/(R*T)
180 IF A$="R" THEN PRINT "EL REDITO=";(1200*I)/(C*T)
190 IF A$="T" THEN PRINT "EL TIEMPO=";(1200*I)/(C*R);" meses"
200 END
```

9.

```
10 REM PITAGORAS
20 PRINT "SI QUIERES CALCULAR LA HIPOTENUSA PULSAR <1>"
30 PRINT "SI QUIERES CALCULAR UN CATETO PULSAR      <2>"
```

```
40 INPUT I
50 IF I<>1 AND I<>2 THEN 40
60 IF I=2 THEN 110
70 INPUT "ENTRAR LOS CATETOS";A,B
80 LET H=SQR(A^2+B^2)
90 PRINT "LA HIPOTENUSA ES=";H
100 GOTO 140
110 INPUT "ENTRAR UN CATETO E HIPOTENUSA";A,H
120 LET B=SQR(H^2-A^2)
130 PRINT "EL OTRO CATETO ES=";B
140 END
```

10.

```
10 REM AREA DEL CUADRADO
20 INPUT "ENTRAR DIAGONAL";D
30 LET L=SQR(D^2/2)
40 PRINT "EL CUADRADO ES DE LADO=";L
50 PRINT "EL AREA DEL CUADRADO ES=";L^2
60 END
```

11.

```
10 REM ECUACION AX 2 + BX=0
20 INPUT "ENTRAR A Y B";A,B
30 LET X=(-B)/A
40 PRINT "X1=";0
50 PRINT "X2=";X
60 END
```

12.

```
10 REM ECUACION AX^2+C=0
20 INPUT "ENTRAR A Y C";A,C
30 IF (-C)/A>0 THEN 60
40 PRINT "LA ECUACION NO TIENE SOLUCION"
50 GOTO 90
```

SIGUE

```
60 LET X=SQR((-C)/A)
70 PRINT "X1=";X
80 PRINT "X2=";X*(-1)
90 END
```

13.

```
10 REM ECUACION BICUADRADA
20 INPUT "ENTRAR A,B Y C";A,B,C
30 LET R=B^2-4*A*C
40 IF R>=0 THEN 70
50 PRINT "ESTA ECUACION TIENE SOLUCION IMAGINARIA"
60 GOTO 110
70 LET X1=((-B)+SQR(R))/(2*A)
80 LET X2=((-B)-SQR(R))/(2*A)
90 PRINT "X1=";X1
100 PRINT "X2=";X2
110 END
```

14.

```
10 REM MEDIA ARITMETICA
20 INPUT "CUANTOS NUMEROS TIENE LA SERIE ";N
30 FOR I=1 TO N
40 INPUT "NUMERO";P
50 LET S=S+P
60 NEXT I
70 LET MEDIA=S/N
80 PRINT "LA MEDIA ARITMETICA ES ";MEDIA
90 END
```

ENUNCIADOS II PARTE

PROBLEMAS DE FISICA

1. Hacer un programa que nos permita calcular la **presión en atmósferas,** o la **fuerza en kilopondios,** o la **superficie en cm^2.** según la formula

$$P = \frac{F}{S}$$

2. Hacer un programa que nos permita calcular la **velocidad media en m/sg,** o el **espacio en m.,** o el **tiempo** en segundos, según la fórmula

$$V = \frac{e}{t}$$

3. Hacer un programa para calcular la **aceleración** dándole la **velocidad inicial,** la **velocidad final** y el **tiempo.**

4. Hacer un programa para calcular la **velocidad final,** dándole la **velocidad inicial,** la **aceleración** y el **tiempo.**

5. Hacer un programa que nos permita **calcular la altura en metros** o el **tiempo en segundos,** que tarda en caer un **peso en el vacío.**

6. Hacer un programa que nos permita calcular el **trabajo en Julios** y la **potencia en Vatios,** dándole la **fuerza en Newtons,** el **espacio en metros** y el **tiempo en segundos.**

7. Hacer un programa que nos permita calcular la **fuerza o el brazo necesarios en una palanca** para levantar un peso dado, con un brazo dado.

8. Hacer un programa para resolver problemas del tipo:

   Se carga una **batería durante x horas con una intensidad de y amperios.** Calcular la carga o cantidad de electricidad almacenada en la batería, expresada en culombios.

9. Hacer un programa que nos permita calcular la **intensidad en amperios, o la diferencia de potencial en voltios, o la resistencia en ohmios; aplicando la ley de Ohm.**

STOP

No pasar la página hasta haber realizado los ejercicios planteados.

SOLUCIONES II PARTE

1.

```
10 REM PRESION FUERZA SUPERFICIE
20 PRINT "SI QUIERES CALCULAR PRESION     <P>"
30 PRINT "SI QUIERES CALCULAR FUERZA      <F>"
40 PRINT "SI QUIERES CALCULAR SUPERFICIE <S>"
50 INPUT A$
60 IF A$<>"P" AND A$<>"F" AND A$<>"S" THEN 50
70 IF A$="P" THEN 90
80 INPUT "ENTRAR PRESION EN ATMOSFERAS ";P
90 IF A$="F" THEN 110
100 INPUT "ENTRAR FUERZA EN KILOPONDIOS ";F
110 IF A$="S" THEN 130
120 INPUT "ENTRAR SUPERFICIE EN CM2 ";S
130 IF A$="P" THEN PRINT "PRESION= ";F/S;" ATMOSFERAS"
140 IF A$="F" THEN PRINT "FUERZA= ";P*S;" KILOPONDIOS"
150 IF A$="S" THEN PRINT "SUPERFICIE= ";F/P;" CM2"
160 END
```

2.

```
10 REM VELOCIDAD ESPACIO TIEMPO
20 PRINT "SI QUIERES CALCULAR LA VELOCIDAD <V>"
30 PRINT "SI QUIERES CALCULAR EL ESPACIO    <E>"
40 PRINT "SI QUIERES CALCULAR EL TIEMPO     <T>"
50 INPUT A$
60 IF A$<>"V" AND A$<>"E" AND A$<>"T" THEN 50
70 IF A$="V" THEN 90
80 INPUT "ENTRAR VELOCIDAD EN m/sg";V
90 IF A$="E" THEN 110
100 INPUT "ENTRAR ESPACIO EN METROS";E
110 IF A$="T" THEN 130
120 INPUT "ENTRAR TIEMPO EN sg";T
```

SIGUE

```
130 IF A$="V" THEN PRINT "VELOCIDAD= ";E/T;" m/sg"
140 IF A$="E" THEN PRINT "ESPACIO= ";V*T;" metros"
150 IF A$="T" THEN PRINT "TIEMPO= ";E/V;" segundos"
160 END
```

3.

```
10 REM ACELERACION
20 INPUT "ENTRAR VELOCIDAD INICIAL EN m/sg ";VI
30 INPUT "ENTRAR VELOCIDAD FINAL EN m/sg ";VF
40 INPUT "ENTRAR TIEMPO TRANSCURRIDO EN sg ";T
50 LET A=(VF-VI)/T
60 PRINT "LA ACELERACION ES IGUAL A ";A;" M/SG2"
70 END
```

4.

```
10 REM VELOCIDAD FINAL
20 INPUT "ENTRAR LA ACELERACION EN M/SG2 ";A
30 INPUT "ENTRAR LA VELOCIDAD INICIAL EN M/SG ";VI
40 INPUT "ENTRAR TIEMPO TRANSCURRIDO EN SG ";T
50 LET VF=A*T+VI
60 PRINT "LA VELOCIDAD FINAL ES IGUAL A ";VF;" M/SG"
70 END
```

5.

```
10 REM CAIDA EN EL VACIO
20 INPUT "Calculo de tiempo <T> Calculo de altura <H> ";A$
30 IF A$="T" THEN 50
40 INPUT "ENTRAR TIEMPO EN SG ";T
50 IF A$="H" THEN 70
60 INPUT "ENTRAR ALTURA EN METROS ";H
70 LET G=9.80
80 IF A$="T" THEN PRINT "EL TIEMPO ES ";SQR(2*h*G);" sg"
90 IF A$="H" THEN PRINT "LA ALTURA ES ";T^2/(G*2);" metros"
100 END
```

6.

```
10 REM TRABAJO POTENCIA
20 INPUT "ENTRAR LA FUERZA EN NEWTONS";F
30 INPUT "ENTRAR EL ESPACIO EN METROS";S
40 LET T=F*S
50 PRINT "EL TRABAJO ES= ";T;" JULIOS"
60 INPUT "ENTRAR EL TIEMPO EN SG ";TI
70 LET P=T/TI
80 PRINT "LA POTENCIA ES= ";P;" VATIOS"
90 END
```

7.

```
10 REM PALANCA
20 INPUT "CALCULA FUERZA -F BRAZO -B";A$
30 IF A$<>"F" AND A$<>"B" THEN 20
40 INPUT "ENTRAR PESO DE RESISTENCIA ";R
50 INPUT "ENTRAR BRAZO DE LA RESISTENCIA ";BR
60 IF A$="F" THEN 80
70 INPUT "ENTRAR FUERZA ";F
80 IF A$="B" THEN 100
90 INPUT "ENTRAR BRAZO DE LA FUERZA ";BF
100 IF A$="F" THEN PRINT "LA FUERZA NECESARIA ES ";(R*BR)/BF
110 IF A$="B" THEN PRINT "EL BRAZO NECESARIO ES ";(R*BR)/F
120 END
```

8.

```
10 REM CALCULO DE CARGA
20 INPUT "HORAS DE CARGA DE LA BATERIA?... ";H
30 INPUT "INTENSIDAD DE CARGA EN amperios; ";I
40 LET Q=H*I
50 LET C=Q*3600
60 PRINT "La electricidad es de: ";C;" culombios"
70 END
```

9.

```
10 REM LEY DE OHM
20 PRINT "Para calcular la diferencia de potencial <V>"
30 PRINT "Para calcular la intensidad              <I>"
40 PRINT "Para calcular la resistencia             <R>"
50 INPUT A$
60 IF A$<>"V" AND A$<>"¡" AND A$<>"R" THEN 50
70 IF A$="V" THEN 90
80 INPUT "Entrar diferencia de potencial en Voltios: ";V
90 IF A$="R" THEN 110
100 INPUT "Entar resistencia en Ohmios: ";R
110 IF A$="I" THEN 130
120 INPUT "Entrar intensidad en Amperios: ";I
130 IF A$="V" THEN PRINT "LA DIFERENCIA DE POTENCIAL: ";R*I" vol."
140 IF A$="R" THEN PRINT "LA RESISTENCIA ES: ";V/I;" Ohmios"
150 IF A$="I" THEN PRINT "LA INTENSIDAD ES : ";V/R;" Amperios"
160 END
```